



NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

**DEVELOPMENT OF A TAILORED METHODOLOGY
AND FORENSIC TOOLKIT FOR INDUSTRIAL
CONTROL SYSTEMS INCIDENT RESPONSE**

by

Nicholas B. Carr

June 2014

Thesis Co-Advisors:

Neil Rowe
Garrett McGrath

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			<i>Form Approved OMB No. 0704-0188</i>	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE June 2014	3. REPORT TYPE AND DATES COVERED Master's Thesis	
4. TITLE AND SUBTITLE DEVELOPMENT OF A TAILORED METHODOLOGY AND FORENSIC TOOLKIT FOR INDUSTRIAL CONTROL SYSTEMS INCIDENT RESPONSE			5. FUNDING NUMBERS	
6. AUTHOR(S) Nicholas B. Carr				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.				
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited			12b. DISTRIBUTION CODE	
13. ABSTRACT (maximum 200 words) <p>This thesis presents a methodology for incident response to identify anomalies and malicious adversary persistence within the networks responsible for the reliable operation of modern society's critical infrastructure. The chapters provide relevant background on the historical development and function of industrial control systems (ICS) and their unique security issues. The study of public technical data from intrusions into control systems produces a set of known adversary tactics for incorporation into the methodology. This work further documents the development of a repeatable technique to collect digital forensic artifacts from production control systems that is compatible with the strict operational constraints of these critical networks. The technique is then applied with a proof-of-concept host-and network-based toolkit for incident response that is tested against real-world data. The goal of the methodology and the supplementary toolkit is to elicit valuable, previously-unavailable findings with which to assess the scope of malicious intrusions into critical ICS networks.</p>				
14. SUBJECT TERMS Industrial Control Systems, Incident Response, SCADA, Digital Forensics			15. NUMBER OF PAGES 99	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UU	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

**DEVELOPMENT OF A TAILORED METHODOLOGY AND FORENSIC
TOOLKIT FOR INDUSTRIAL CONTROL SYSTEMS INCIDENT RESPONSE**

Nicholas B. Carr
ICS-CERT Chief Technical Analyst, Department of Homeland Security
B.S., Lehigh University, 2006 and 2007

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN CYBER SYSTEMS AND OPERATIONS

from the

**NAVAL POSTGRADUATE SCHOOL
June 2014**

Author: Nicholas B. Carr

Approved by: Neil Rowe
Thesis Co-Advisor

Garrett McGrath
Thesis Co-Advisor

Cynthia Irvine, Ph.D.
Chair, Department of Cyber Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis presents a methodology for incident response to identify anomalies and malicious adversary persistence within the networks responsible for the reliable operation of modern society's critical infrastructure. The chapters provide relevant background on the historical development and function of industrial control systems (ICS) and their unique security issues. The study of public technical data from intrusions into control systems produces a set of known adversary tactics for incorporation into the methodology. This work further documents the development of a repeatable technique to collect digital forensic artifacts from production control systems that is compatible with the strict operational constraints of these critical networks. The technique is then applied with a proof-of-concept host- and network-based toolkit for incident response that is tested against real-world data. The goal of the methodology and the supplementary toolkit is to elicit valuable, previously-unavailable findings with which to assess the scope of malicious intrusions into critical ICS networks.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I.	INTRODUCTION.....	1
A.	BACKGROUND	1
B.	MOTIVATION	1
C.	PROBLEM STATEMENT	2
D.	APPROACH AND LIMITATIONS.....	3
	1. Scope.....	3
	2. Environment and Data Availability	3
E.	THESIS ORGANIZATION AND CONTRIBUTIONS	3
II.	BACKGROUND	5
A.	INDUSTRIAL CONTROL SYSTEMS	5
	1. Industrial Control Systems History	5
	2. Industrial Control Network Composition	7
	3. ICS Domain Interconnection Methods	10
	4. ICS Communication Protocol Inspection	11
B.	INDUSTRIAL CONTROL SYSTEMS SECURITY.....	13
	1. Unique ICS Security Vulnerabilities.....	13
	a. <i>Legacy Equipment Challenges</i>	13
	b. <i>Real-time Availability Requirements</i>	15
	c. <i>Problematic Patching</i>	15
	2. Consequences of ICS Failure	16
III.	CASE STUDY TECHNICAL ANALYSIS.....	19
A.	CATEGORIES OF MALICIOUS ACTIVITY.....	19
	1. Custom Targeted Exploits.....	19
	2. Commodity Malware in the ICS Domain	21
	3. Unauthorized Persistent ICS Network Access	24
B.	KEY POINTS AND TRENDS FROM ICS INCIDENTS	26
IV.	METHODOLOGY	29
A.	FRAMEWORK FOR IDENTIFYING MALICIOUS ACTIVITY.....	29
	1. Framework Overview	29
	2. Framework Constraints	30
B.	TOOLKIT SELECTION	32
	1. Host-based Tools	32
	2. Network-based Tools	34
	3. Toolkit Coverage.....	35
C.	IDENTIFICATION OF ADVERSARY TACTICS.....	36
	1. ICS Field Device Anomalous Operations	36
	2. External Network Communication	39
	3. Registry, Startup, and Scheduled Task Persistence	43
	4. Process Injection and Hijacking.....	45
	5. File System Sabotage	46
	6. Internal Network Lateral Movement.....	48

7.	Portable Media Exploitation.....	49
8.	User Activity Abnormalities and Remote Access Abuse.....	50
D.	CONSOLIDATED TACTIC IDENTIFICATION	52
V.	EXPERIMENTS	55
A.	DATASETS	55
B.	INITIAL RESULTS	55
1.	Operation within Constraints	56
2.	Explanation of Host-based Toolkit Functions.....	57
3.	Concerning Findings.....	60
4.	Valuable Discoveries	63
C.	TOOLKIT LIMITATIONS	67
1.	Toolkit Deficiencies.....	67
2.	False Positive Errors.....	69
3.	False Negative Analysis	70
VI.	CONCLUSIONS AND FUTURE WORK.....	71
A.	IMPACT	71
B.	FUTURE WORK.....	71
C.	CONCLUDING REMARK.....	73
	LIST OF REFERENCES.....	75
	INITIAL DISTRIBUTION LIST	83

LIST OF FIGURES

Figure 1.	General ICS Network Composition Diagram, from [6].....	8
Figure 2.	Abstracted ICS Network Diagram, from [10].....	9
Figure 3.	ICS Malicious Activity Identification Methodology	29
Figure 4.	Comparison of IT and ICS Penetration Testing Actions, from [58]	31
Figure 5.	WMIC usage on Siemens SIMATIC WinCC HMI, from [62].....	33
Figure 6.	Modbus and DNP3 Bro log fields.....	34
Figure 7.	Host-based passive ICS device enumeration batch script.....	38
Figure 8.	External ICS network communication identification Bro script excerpt.....	41
Figure 9.	Bro script variable translation to jVectorMap overlay data.....	42
Figure 10.	Excerpt from this thesis' ICS process and service attribute whitelist.....	46
Figure 11.	Host-based scripts running at lowest priority	56
Figure 12.	Host-based command usage and operating system compatibility	58
Figure 13.	Identified communication paths from ICS network.....	61
Figure 14.	External connections output from this thesis' network-based toolkit.....	61
Figure 15.	USB insertion script output on test system	62
Figure 16.	Host characteristics extracted from command-line utilities.....	63
Figure 17.	Summary of host characteristics extracted from network data	65
Figure 18.	ICS protocol frequency analysis in dataset	66
Figure 19.	Three command-line tool queries with mutually-exclusive results	68
Figure 20.	Content distribution of initial ICS-specific whitelist.....	73

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACL	access control list
APT	advanced persistent threat
ARP	address resolution protocol
ASN	autonomous system number
BGP	border gateway protocol
C2	command and control
CIP	common industrial protocol / critical infrastructure protection
CRC	cyclic redundancy check
DDoS	distributed denial-of-service
DHCP	dynamic host configuration protocol
DLL	dynamic link library
DNS	domain name system
DHS	Department of Homeland Security
DNP3	distributed network protocol revision 3
Ethernet/IP	Ethernet industrial protocol
FTP	file transfer protocol
GE	General Electric
GUI	graphical user interface
HMI	human-machine interface
HTTP	hypertext transfer protocol
I/O	input/output
ICCP	inter-control center communications protocol
ICMP	Internet control message protocol
ICS	industrial control systems
ICS-CERT	Industrial Control Systems Cyber Emergency Response Team
IDS	intrusion detection system
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet protocol
IT	information technology
JSON	JavaScript object notation

LAN	local area network
MAC	media access control
MBR	master boot record
MS-DOS	Microsoft Disk Operating System
NERC	North American Electric Reliability Corporation
NIST	National Institute of Standards and Technology
NTP	network time protocol
NTSB	National Transportation Safety Board
NRC	Nuclear Regulatory Commission
ODVA	Open DeviceNet Vendors Association
OUI	organizationally unique identifier
PCAP	packet capture
PLC	programmable logic controller
RAT	remote administration tool / remote access Trojan
RDP	remote desktop protocol
ROM	read-only memory
RPC	remote procedure call
RTU	remote terminal unit
SCADA	supervisory control and data acquisition
SPAN	switch port analyzer
SQL	structured query language
SSH	secure shell
TCP	transmission control protocol
TFTP	trivial file transfer protocol
TTPs	tactics, techniques, and procedures
UDP	user datagram protocol
USACE	United States Army Corps of Engineers
USB	universal serial bus
VNC	virtual network computing
VPN	virtual private network
WMIC	Windows Management Instrumentation Command-line

ACKNOWLEDGMENTS

My sincere gratitude goes to Dr. Irvine, Dr. Davis, and the entire Cyber Academic Group at the Naval Postgraduate School for their guidance, encouragement, and unrelenting pressure that made this thesis possible. To my advisors—thank you for your patience and for spending the time to ensure that I succeeded. To my fellow colleagues within the Department of Homeland Security’s pilot cohort for this challenging program—a few of us made it to the finish line! To my wife and family—thank you for your unwavering support and daily motivation through all the late nights and weekends I spent in front of my computer... see—I really was working on something!

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

Throughout history, malicious attackers have targeted critical infrastructure assets that provide nations with essential services. Around 600 BC, Solon of Athens contaminated the water supply for the town of Cirrha, allowing Athens to swiftly conquer the town while the Cirrhaens were violently ill [1]. Two thousand six hundred years later, Vitek Boden of Brisbane maliciously released thousands of gallons of raw sewage into waterways of Maroochy Island, causing significant financial and economic damages and forcing the residents to relocate [2], [3]. In both cases, a single malicious actor obtained unauthorized access to critical infrastructure assets and, prior to detection, caused severe physical consequences. Now critical infrastructure systems are more susceptible to malicious attacks than ever before.

The reliable operation of modern society's critical infrastructure depends on industrial control systems (ICS), the embedded software systems that allow an operator or device to monitor and control industrial processes. These automation systems are ubiquitous and heterogeneous; they were designed with much implicit trust and are not very compatible with most modern security solutions. In 2010, Stuxnet demonstrated that traditional network protections like segmentation and intrusion detection systems (IDS) alone are insufficient in securing control systems [4]. Insecurity of these devices can lead to severe consequences and malicious hacking tools can be effective without much difficulty to an attacker. To successfully leverage these tools, an adversary must access the trusted network, leaving specific forensic artifacts and indicators of ICS malicious activity.

B. MOTIVATION

An ICS cyber incident response process is not well developed and we lack tools built specifically for identifying current or historical adversary presence within the critical systems domain. Few published efforts reveal actionable technical solutions for ICS security practitioners and none focus on reliably identifying malicious, persistent

access within live data from production ICS devices. The primary motivation for this thesis was based on technical gaps observed and reported [5] during the course of ICS security assessments and cyber incident response.

Previous studies have outlined the need for forensic collection abilities within the ICS environment. The Department of Homeland Security (DHS) Industrial Control System Cyber Emergency Response Team (ICS-CERT) has concluded that traditional forensic tools are not suitable for use in ICS networks [6]. ICS-CERT provides a high-level strategy for forensics, makes the case for having an incident response capability, and presents a breakdown of what is different about ICS systems. However, the ICS-CERT best practice documents only recommend that the capability should be created but the papers do not offer specific tools and techniques to implement incident response. Some research has posited that live forensics on automation networks is feasible; however no specific adversary identification techniques have been provided and no available forensic toolkits are available that have been designed for use on ICS networks [7].

C. PROBLEM STATEMENT

Critical infrastructure owners and operators are poorly equipped to discover and respond to intrusions into ICS networks. Effective, compatible tools do not exist to reliably extract the necessary technical data to analyze ICS environments with modern incident response techniques [7]. Adversaries construct surreptitious pathways to their target systems that provide persistent access for reconnaissance and future malicious activity. These adversary persistence mechanisms often remain undetected for significant periods of time. Security teams require a repeatable, tailored response methodology employing host and network data collection and analysis techniques to identify malicious pathways and adversary presence on ICS networks. The methodology and supplementary toolkit within this thesis represent a step toward addressing that need.

D. APPROACH AND LIMITATIONS

This thesis proposes a structured methodology to identify malicious activity by using host-based forensics and network analysis to identify anomalous client-side attack vectors during ICS assessments and incident response.

1. Scope

This thesis will cover analysis of control systems that respond to command-line forensic interrogation techniques and that communicate over accepted, interpretable ICS protocols. It will not cover embedded device firmware extraction or offline drive image analysis within the host-based methodology. ICS field devices will be discussed, but direct input/output (I/O) protocols and serial communication are not. This thesis also does not cover response actions should adversary presence be detected, nor disaster recovery, although specific case-by-case suggestions will be made.

2. Environment and Data Availability

To develop the most robust and reliable methodology, real data from ICS networks will be used in this study. Some experimentation may be conducted through the course of regular assessments and incident response with critical infrastructure asset-owner approval. Multiple regulations protect disclosure of this data; anonymized data will be used when possible, to include host-based artifacts and scrubbed ICS network traffic. Additional experiments will be conducted within a closed network that replicates real ICS architectures.

E. THESIS ORGANIZATION AND CONTRIBUTIONS

Chapter II provides background on ICS systems and security. Chapter III contains technical case studies of relevant cyber incidents. Chapter IV describes a methodology for identifying malicious activity. Chapter V presents the experiments and findings of this study. Chapter VI provides a summary and recommends areas for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND

A. INDUSTRIAL CONTROL SYSTEMS

The reliable operation of modern society's critical infrastructure depends on the proper functioning of industrial control systems. These vital systems are widely implemented across all infrastructure sectors to automate large industrial processes.

Components of ICS networks are designed to allow embedded logic to control a process efficiently without constant human intervention, and thus these components have specific roles and constraints to operate as low-level building blocks for industrial automation.

1. Industrial Control Systems History

In 1959, a Texaco refinery in Port Arthur, Texas installed the first ICS device in history, a Thompson Ramo Wooldridge RW-300 direct control process computer [8]. Control systems were originally designed to monitor and control industrial processes in complete isolation, much like server mainframes were initially configured. One central master unit provided all computing, control, and monitoring functionality, quietly executing simple instructions or ladder logic [9]. Fault tolerance was achieved through complete system redundancy: A duplicate ICS master unit provided all functions of the original and monitored the primary ICS system's operation. If the secondary system detected a fault, it commandeered all operations. Vendors continued to follow this mainframe architecture configuration for ICS systems through the 1960s and continuing into the early 1980s, installing unique systems with proprietary protocols as automation technology matured.

As personal computer (PC) costs became less prohibitive and Local Area Network (LAN) technology was embraced for business networks in the late 1980s, individual computers began to replace static ICS components which enabled distributed functionality and processing across multiple control systems [9]. Data historians and human-machine interfaces (HMIs) were implemented not as standalone systems but as vendor-specific proprietary software on specialized personal computers. The data

historian is a database that contains a history of the state of a process-control system [10]. HMIs still provide the graphical user interface (GUI) front-end for operators today and continue to be one of the few human-to-machine touchpoints in the largely automated control environment. These computer-based processes ran modified versions of common operating systems and were generally vendor-provided systems, with only the vendor able to provide upgrades and system maintenance. As a result of the newly distributed architecture, the ICS domain no longer required full-time redundant failover systems because the PC-based ICS systems could share the operations load of failed components.

In the late 1990s, vendors started to embrace commercial off-the-shelf computer systems and networking hardware. These vendors began to adapt their legacy proprietary protocols for field devices so that the systems could communicate over the Internet protocol suite (TCP/IP). The last section of this chapter examines ICS protocols more comprehensively. Around this time, Microsoft Windows became the operating system of choice for HMIs, engineering workstations, and several ICS servers. Vendors used old or modified versions of Windows that, once initially tested for compatibility with vendor equipment, were rarely updated.

These newly networked and communicating field devices, like remote terminal units (RTUs) and programmable logic controllers (PLCs), allowed for customized implementations. RTUs are field devices that transmit telemetry data (information that is collected at remote or inaccessible points) to master systems. RTUs also accept commands from those master systems to control connected objects. PLCs are also field devices and they execute a wide array of programmable functions such as vibration monitoring, catalyst loading, area monitoring, and product loading/unloading. PLCs generally execute ladder logic but now can run higher level programming languages like C++; they offer the lowest-level code before reaching physical systems. With the introduction of these field devices, the ICS component manufacturers enabled the system-integrators and critical infrastructure companies to make their own modifications and leverage their existing network infrastructure. Companies have continued to develop networked ICS architectures that allow for streamlined performance tracking, accurate billing, and off-site backup capabilities, despite the growing security concerns that this

interconnection has introduced. This fully distributed and networked architecture is necessary for industrial applications that require distributed monitoring points, such as electric power transmission and distribution, oil and gas pipeline and production operations, and water utility operations.

The centralized consolidation, processing, and management of the field devices spawned another system term “SCADA” which is often incorrectly used to represent the larger set of ICS components. Supervisory control and data acquisition (SCADA) systems extend the capabilities of an automated system so that it can be monitored and even controlled from a remote location [11]. The implementation of SCADA systems allows for the control of industrial processes across widespread geographical locations and facilitates the fetching and presentation of current data values to a human operator [12].

2. Industrial Control Network Composition

All automation components that have been developed over the past fifty years (SCADA, HMIs, RTUs, and PLCs) are implemented in some form in most ICS network configurations. SCADA systems continue to manage very large-scale processes at multiple sites and over large distances. HMIs are the devices which present process data to their human operators, who control and monitor the processes. Some common HMIs in industry include Wonderware, Siemens WinCC, Rockwell RSView, and Areva’s e-terra [10]. RTUs interconnect all the sensors in the process and also convert those sensor signals to digital data, which is routed to the SCADA system. PLCs are field devices that are cheap, flexible, and highly-configurable. When combined within an industrial control network, these devices retain features of the individual system, leading to unique, expensive configurations that have the functionality and weaknesses of the legacy equipment.

ICS components can be interconnected with a variety of mediums. Direct-wiring is common for intra-facility connections. Microwave communications backbones are very frequently used between facilities, especially in oil and natural gas pipelines, but fiber optic inter-facility deployments are becoming more common. Both spread-spectrum and

narrow-band radio are used to connect remote ICS components, however dial-up or cellular modems continue to provide connection to these systems.

As explained above, there are many components and countless possible configurations. Attempting to broadly capture several possible ICS network architectures produces a complex diagram, such as Figure 1.

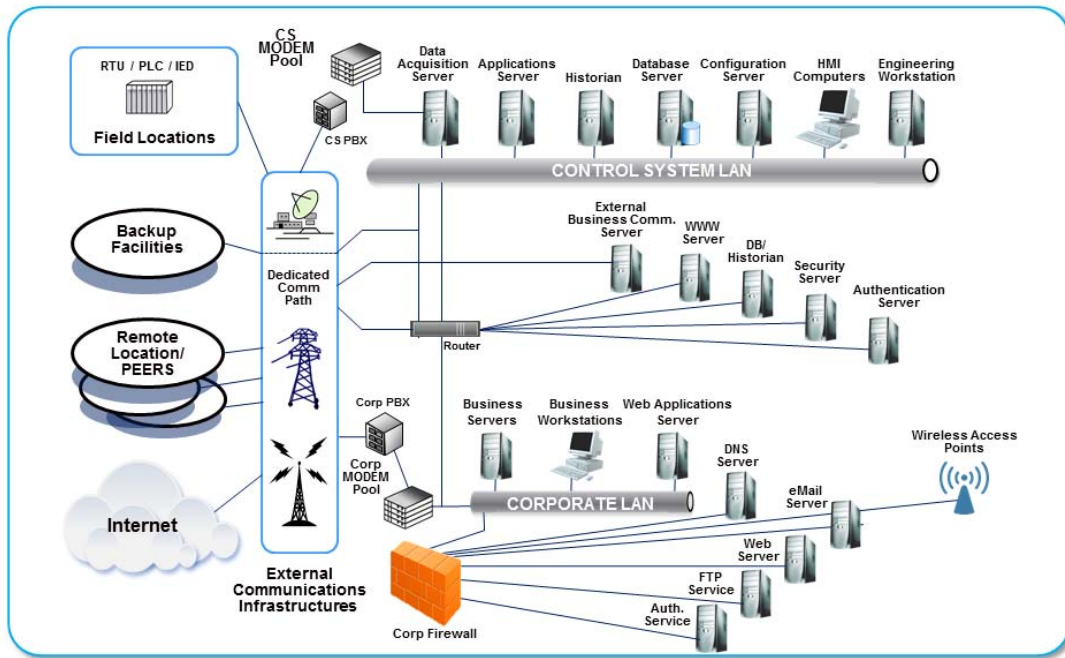


Figure 1. General ICS Network Composition Diagram, from [6]

While this is helpful for understanding the complexities of these systems, a more abstracted model can be used to encapsulate the myriad of components and configurations. Researchers introduced an abstracted ICS architecture, shown in Figure 2, which identifies the key functions within the ICS domain [10].

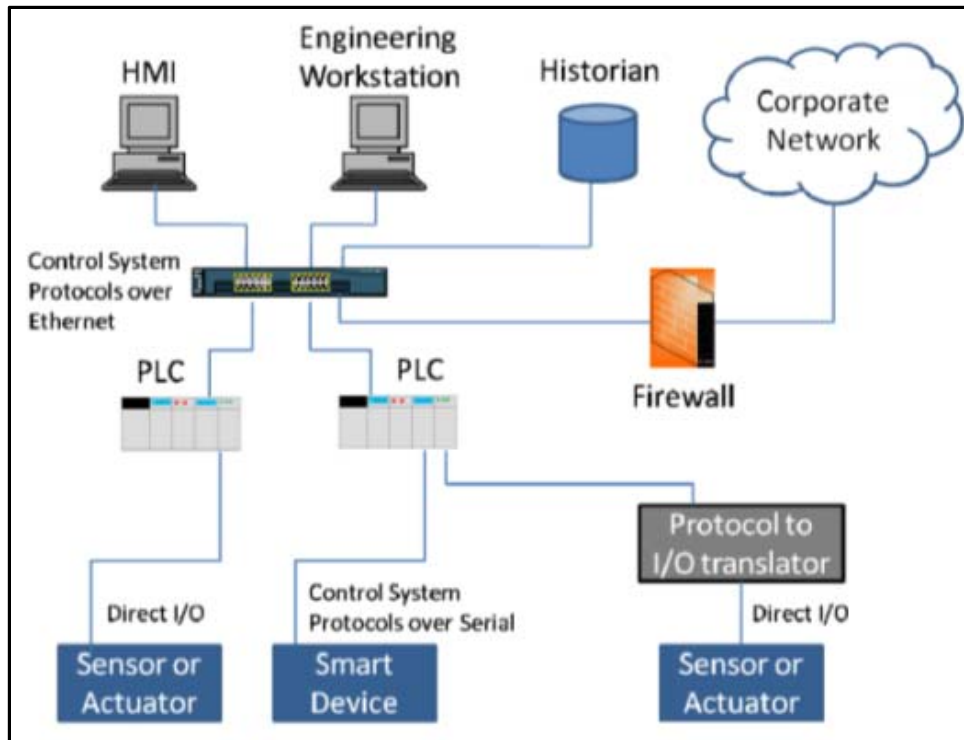


Figure 2. Abstracted ICS Network Diagram, from [10]

At the top right of Figure 2, the traditional IT network, which hosts the corporate network and controls site manufacturing operations, is interconnected to the rest of the ICS network by one of several domain-interconnection methods discussed in the next subsection. Inside the firewall in Figure 2, the control center network houses the supervisory services that reside on the engineering workstation, SCADA input/output (I/O) server, and the HMI. These supervisory control systems connect using ICS protocols over Ethernet to the field devices such as PLCs or RTUs that are often located at distributed sites. The field devices receive commands from the supervisory systems that instruct the field devices to subsequently control or acquire data, often over direct I/O, from the physical ICS assets like mechanical valves, circuit breakers, voltage regulators, digital temperature sensors, or other smart devices. The data acquired from the physical ICS assets is transferred from the PLCs and RTUs to a central control center where it is displayed to the operator using an HMI.

3. ICS Domain Interconnection Methods

In addition to the distribution and communication paths between ICS components, there are often additional network interconnections between the traditionally isolated ICS domain and the corporate network. These interconnections are introduced for a variety of reasons. The ICS network contains valuable information for business applications, such as billing and financial data, equipment trending, and operational reports. Interconnections to external networks may have been introduced to allow for remote vendor support or, in the case of the inter-control center communications protocol (ICCP), for utilities to share electrical power status for grid stability [14]. Remote access technology allows for access to corporate software from field locations and provides capability to manage devices that are difficult to access.

Dedicated lines are expensive, while the Internet is cheap and pervasive. According to the National Institute of Standards and Technology (NIST), “widely available, low-cost Internet protocol (IP) devices are now replacing proprietary solutions, which increases the possibility of cybersecurity vulnerabilities and incidents” [15]. Low cost and easy-to-install-and-maintain wireless connectivity has been added to field devices, allowing for the bypass of the physical security boundary and direct connection of field devices to the Internet.

ICS inter-domain networking can be architected using a variety of methods, such as explicit direct connections, firewall-controlled connections, demilitarized zone (DMZ) only connections, or data diodes. Direct connections can be hard to trace as they are constructed by employees or vendors uniquely for the site, usually with standard protocols such as secure shell (SSH), Telnet, file transfer protocol (FTP), virtual private network (VPN), or with dial-up connections. This means that the direct connections may not be known to the security team. Firewalls and access control lists (ACLs) are often used for this interconnection and only allow certain types of connections; but the software for them provides only limited support for ICS protocols. DMZs are common with these interconnections; however they should not be constructed to have access to the Internet like traditional web server DMZs. When properly implemented, isolated networks can communicate with the DMZ but not with one another, making the DMZ

good for storage servers such as databases and data historians. Data diodes enforce unidirectional network flow, making them ideal for the ICS environment, but they do not allow for acknowledgement or response and thus are not fully compatible with the transmission control protocol (TCP). This means that data diodes, although difficult to implement with many protocols, can address a limited set of inter-domain communication needs. Regardless of interconnection type, a single control such as a DMZ, firewall, or data diode cannot provide sufficient defense alone. Since there are no guaranteed defenses for business networks and almost all are interconnected with ICS networks in some way, business network incidents can intentionally or collaterally affect ICS networks.

It is believed that, in 2013, part of the Austrian and German electric power grid almost failed due to a single misconfigured ICS network interconnection when a status request command packet, sent from a German gas company as a test of their new equipment, made its way onto an Austrian energy power control and monitoring network [16]. Once on the Austrian ICS network, the message generated thousands of reply messages and flooded the control network. In order to resolve the self-inflicted distributed denial-of-service (DDoS) incident prior to power outages, a portion of the monitoring and control network had to be isolated and disconnected [16]. Thus, it is clear that not only the interconnection medium must be understood but also the communication protocols used by the ICS devices.

4. ICS Communication Protocol Inspection

Since the development of ICS technology was vendor-driven with a wide variety of competing hardware, software, and capabilities, the communication technology was similarly developed in a loose, ad-hoc fashion and lacked a central standards body. The resulting list of communication protocols include Modbus and the distributed network protocol revision 3 (DNP3) as well as hundreds of proprietary protocols like ANSI X3.28, CDC Types 1 and 2, Conitel 2020/2000/3000, DCP 1, Gedac 7020, IBM 3707, ICCP, IEC 61850, Landis & Gyr 8979, OPC, Redac 70H, Tejas 3 and 5, TRW 9550, and

UCA. In total, there are estimated to be between 150 and 200 different ICS protocols [17].

Despite the fragmented proprietary protocols and slow adoption rate of new technology in ICS, Modbus, DNP3, and the Ethernet industrial protocol (Ethernet/IP) are the most-used ICS protocols [18]. The Modbus messaging protocol was developed by Modicon in 1979 to establish client-server/master-slave communication between intelligent devices, first over serial lines then later incorporating TCP [19] (the TCP suite are the most common protocols on the Internet). DNP3 was developed as a set of protocols for data acquisition and control equipment communication, and in 2010 the Institute of Electrical and Electronics Engineers (IEEE) established DNP3 as the standard for electrical power system communications (IEEE 1815) [20]. Lastly, Ethernet/IP was originally developed by Rockwell Automation in 2001 but is now managed by the Open DeviceNet Vendors Association (ODVA) and is an application-layer protocol that implements the common industrial protocol (CIP) over TCP [21].

ICS protocols have multiple telemetry schemes such as reporting by exception (which is common in Europe), in round-robin communications, or at a time polling interval. Most of these protocols are primitive and field devices cannot be reliably queried to see what protocols they support. All these factors result in a highly-complex forensic and incident response process. There is a recent trend toward routable, industry-standard protocols that may someday replace these legacy vendor-specific proprietary protocols [15]; however the long product life cycle and high cost for replacing ICS components means this standardized environment is a long way off.

Despite their differences, every ICS protocol functions using master-slave communication. The master polls for data, controls slave devices, and maintains a repository of data. The slaves, transmitting either by polling or reporting by exception, respond to master commands. Although all components in the ICS domain are either a master or a slave, it is important to note that slaves can have more than one master, and a device can be a master in one environment and a slave in another as in a tiered architecture. This tiered structure, with a master at a remote site gathering all data for transmission to the next hop, saves bandwidth and reduces the poll cycle. Also important

to this study is that most modern protocols researched communicate, albeit via simple wrappers and often in cleartext, using TCP/IP for subprotocols, which allows for some level of passive network traffic parsing.

B. INDUSTRIAL CONTROL SYSTEMS SECURITY

As explained in the previous section, ICS design allows operators to interact with the control systems at an abstract level, significantly lowering the technical expertise and knowledge of the system required to keep the process running. The resulting state of ICS network security is that field devices are low capability, designed for performance rather than security, and operate on a large number of unique protocols that must all be protected equally [17]. ICS systems were built to be highly available machines and integrity and confidentiality were afterthoughts. That is, ICS hardware was never designed with security in mind because they were originally deployed in isolated environments, removed from external networks. As those systems have become increasingly connected to the network across much of critical infrastructure, ICS systems are more accessible and susceptible to malicious attacks. “Control systems are the ‘brains’ of the control and monitoring of the bulk electric system and other critical infrastructures, but they were designed for functionality and performance, not security. Most control systems assume an environment of complete and implicit trust” [23].

1. Unique ICS Security Vulnerabilities

The same control systems that replaced so many human functions, such as flipping a switch or turning a knob based on automated thresholds, also introduced a range of security concerns. The major vulnerabilities for ICS are in three areas: the prevalence of legacy equipment, the requirement for real-time availability, and the patching difficulties associated with partially or fully segmented networks.

a. Legacy Equipment Challenges

The product life cycle for ICS equipment is considerably longer than traditional information technology (IT) systems. 20-year-old systems are common compared to 3-to-5-year-old systems found in traditional IT networks. Due to the cost of individual

components, it is not feasible to replace all insecure legacy equipment and these challenges are common across all of critical infrastructure's ICS deployments. Vendor support is limited compared to traditional computers with few support styles and often a single vendor supporting many systems. Forensic capabilities are lacking on most ICS equipment since field devices do not generally store logs and their alarms and responses can be suppressed.

To communicate in these mixed environments, with both legacy and modern equipment, proprietary ICS protocols were modified to be used in IP-based networks. The result is already weak protocols transformed into cleartext packets wrapped in TCP/IP layers, which has only increased the pre-existing attack surface. With prevalent cleartext communications and limited authentication or validation, ICS networks provide ample opportunity for tampering, interception, and injection of data. With persistent access to the ICS supervisory local area network, an adversary could conduct an eavesdropping attack on the SCADA server's communication with PLCs, since this is often in cleartext and could be manipulated. This could enable injected telemetry data, adversary knowledge of system events, enumeration of equipment, or loss of market-relevant information. From a security perspective, most ICS traffic is accepted if addresses match and cyclic redundancy checks (CRCs) validate.

Modern solutions like encryption are often not implemented and frequently are not supported within legacy ICS protocols. In fact, the popular Modbus and DNP3 protocols currently do not support authentication, integrity checking, authorization, or encryption. Third-party security solutions may not be applicable since the components were designed to support the intended industrial process and the components may not have enough computing resources or memory to support the addition of security capabilities. The later that security is considered in the development of many devices, the more difficult and expensive its security can become. This problem is expected to continue for some time, since the long lifespan of ICS components requires that even new equipment added to the network will have to be backwards-compatible with technology or protocols that are 10-or-more years old [24].

b. Real-time Availability Requirements

Any solution attempting to address the legacy equipment vulnerabilities must not introduce latency or overhead into the network traffic, especially 0% downtime critical traffic that must be operational 24 hours a day for 365 days a year.

An ICS network has minimal tolerance for communication delay or data loss. The environment is expected to be available for extended periods of time and to meet strict timing requirements [14]. The “flipped CIA triad” is often referenced by ICS security experts [25]. Whereas traditional IT systems are built around confidentiality, integrity, and availability (CIA triad), in that order, control systems are the opposite. They were designed with the requirement that real-time availability comes first, then the integrity of the telemetry data, and lastly confidentiality.

Real-time availability requirements were historically addressed with redundancy [25] but full-time redundant backups are no longer the industry standard. Additionally, increased competition within multiple critical infrastructure industries and a focus on cost control has led to increased asset use with fewer trained staff responsible to maintain the systems’ uptime.

With the same uptime requirements but without fully redundant backups, simple IT functions like rebooting may not be acceptable due to process availability requirements. The throughput and uptime requirements of these networks create an environment where continuous reliable operations will always trump security assessments, forensics, and incident response.

c. Problematic Patching

ICS components tend to be inadequately patched compared to IT systems. The security afforded by any amount of ICS network segmentation is exchanged for the ability to easily manage patches, antivirus definitions, and firmware updates. Patch management is essential to update or repair components of systems that have identified vulnerabilities affecting the validity and integrity of device operation. However, patching of ICS software in critical infrastructure is “inconsistent at best and non-existent at worst” according to NIST [15]. This is compounded by the lack of vendor support and

the difficulty of upgrading within the highly volatile and sometimes unstable environment in which control systems reside. Months of planning is often required in order to take offline and apply patches to production ICS systems. Critical infrastructure asset operators must weigh the planning difficulties and costs of patching ICS systems for stable and accurate operations against the possibility of malicious tampering with that component. 87 percent of ICS-specific vulnerabilities reported to DHS in 2013 were exploitable remotely over the network [26]. Furthermore, even if patches are scheduled and applied properly, they can introduce instability into the ICS domain if not thoroughly tested.

This means that a growing number of “forever day” vulnerabilities are being discovered in older control systems [27]. These vulnerabilities, unlike zero-day exploits where vendors and security firms can deploy patches, exist on the legacy equipment described above and can be targeted specifically with the knowledge that they are weaknesses on older control systems that are not continually supported.

2. Consequences of ICS Failure

NIST states that, “ICS are typically used in industries such as electrical, water and wastewater, oil and natural gas, chemical, transportation, pharmaceutical, pulp and paper, food and beverage, and discrete manufacturing (e.g., automotive, aerospace, and durable goods)” [15]. These systems directly control the processes that operate and stabilize our critical infrastructure, including everything from dams and water treatment plants to electric power utilities and nuclear generation plants. The failure of control systems can directly manifest in the physical world with effects ranging from regulatory non-compliance to severe physical disasters.

Since ICS systems are crucial to the operation of the electric grid, their failure could lead to blackouts, economic disruption, and loss of life [15]. In the electricity sector, instability can lead to cascading outages and loss of communication, especially power systems with dynamic and dramatic changes in load. In that scenario, there is a complex, multi-step restoration process. First, critical loads like hospitals must have power restored, then generation facilities, transmission lines, distribution feeders, and

lastly corporate and residential power service. In the oil and natural gas industry, system failure can result in fuel shortages with economic impacts and even explosions, causing loss of human life or environmental catastrophes. In the water sector, improper treatment or contamination of water, release of waste, or reduced pressure to emergency systems like fire hydrants are possible consequences. Chemical ICS systems can be tampered with resulting in chemical formula manipulation. Even the transportation sector relies heavily on ICS networks for control of mass transit, and their failure could lead to train derailments or crashes.

The hardware and software for ICS components is almost exclusively foreign-owned [29] and there are a limited number of critical asset manufacturers, with long lead times required for high-value components. For example, it can take six months to receive a replacement for certain transformers. The United States' critical infrastructure is often referred to as a "system of systems" because of the interdependencies that exist between its various industrial sectors as well as interconnections between business partners [30], [31].

The failures of these systems can be very real. In June 1999, the Olympic Pipeline Company's pipeline ruptured causing massive physical consequences, including three deaths and multiple additional injuries including hydrocarbon inhalation poisoning [32]. The National Transportation Safety Board (NTSB) concluded that the SCADA services failed on its single Ethernet backbone and event logs indicated that the SCADA system failed to execute the command, partially as a result of database development work being done on the SCADA system while it was being used to operate the pipeline, which led to the systems non-responsiveness during critical operations [32]. Had the supervisory ICS components remained responsive to the commands of the Olympic controllers, the controller operating the accident would have been able to initiate actions that would have prevented the pressure increase that ruptured the pipeline.

In August 2003, the northeastern region of the United States and Canada experienced a blackout that affected 50 million people due to cascading failures of electrical grid operation. While the root cause of the blackout was tree limbs contacting transmission wires, it was later determined that a previously unknown software bug in

GE Energy's XA/21 system at FirstEnergy Corporation in Akron, Ohio was partially responsible for the initial breakdown when it silently failed and did not trigger the alarm system [33].

In March 2008, an engineer at the Hatch nuclear power plant installed a software update on a computer operating within the plant's business network. When that corporate network computer rebooted, it reset the data on the control system, causing safety system to misinterpret the lack of data as a drop in coolant water reservoirs and a plant shutdown automatically began [34]. The nuclear plant shutdown took 48 hours to recover.

III. CASE STUDY TECHNICAL ANALYSIS

A. CATEGORIES OF MALICIOUS ACTIVITY

To identify malicious activity in the ICS domain, it is necessary to review case studies of previous intrusions. Due to the limited availability of public data on past ICS incidents, this thesis categorizes similar events together to isolate behavioral trends. Past malicious activity related to control systems can be assigned to one of three categories: custom targeted exploits, commodity malware in the control environment, and unauthorized persistent ICS network access.

1. Custom Targeted Exploits

This category of malicious activity includes nation-state-sponsored hacking or other highly-targeted and well-funded intrusions, often using custom exploits. These attacks have targeted PLCs and other ICS field devices and can involve firmware tampering or exploiting other device vulnerabilities. Targeted attacks such as those sponsored by nation states tend to use zero-day exploits that have not been publicly disclosed or patched by the vendor. Historical examples of targeted critical infrastructure attacks include the proof-of-concept Aurora exploit, Shamoon, and most notably Stuxnet.

In early 2003, a marine terminal in Venezuela was the target of attempted sabotage. The technical details have not been shared publicly, but it is believed that a team of hackers obtained access to the SCADA network of the oil tanker loading machinery and overwrote PLCs with an empty program module [16]. The result was an eight hour halt of the oil tanker loading process until the backup ladder logic was reinstalled on the PLCs.

In September 2007, DHS demonstrated the feasibility of customized targeted attacks on ICS systems with the Aurora exploit [35]. Aurora damaged rotating electrical equipment with multiple torque shocks by opening a breaker then closing a generator back into the power system out of phase [36]. This electrical attack could be conducted locally or remotely using unauthorized access to conduct man-in-the-middle or address resolution protocol (ARP) cache poisoning to inject breaker trip commands. Although

several mitigation techniques currently exist [37], Aurora proved empirically the ability to attack a physical device via the Internet and underscored the critical need to identify malicious access to ICS assets.

In June 2010, a piece of malware was uncovered by researchers from Belorussian security firm VirusBlokAda that would eventually become known as the Stuxnet worm. Stuxnet exploited four zero-day vulnerabilities and used two compromised digital certificates. The malware did not target the PLCs and field devices directly; it exploited high-level application software on the controlling Windows-based systems with the ability to program PLCs [4]. The PLC rootkit code resided on and was executed on the engineering workstation and not the PLC itself [38]. The worm affected Windows 2000, Windows XP, Windows Server 2003, Windows Vista, and Windows 7 workstations. In addition to probable initial universal serial bus (USB) device infection, Stuxnet spread over the LAN using network shares [39]. Researchers have concluded that Stuxnet code remained idle for an average of 12.8 days after initial infection before executing, then persisted quietly for 26.6 days between subsequent executions [4].

Although data deletion attacks have existed since at least 1998, when the CIH virus was created to overwrite a portion of the hard disk as well as the computer's flash read-only memory (ROM) [40], the recent resurgence of these attacks has renewed concern over this style of attack. In August 2012, the Shamoon data deletion attack targeted the world's largest oil company, Saudi Aramco. ICS-CERT claims that a destructive attack similar to Shamoon could just as easily have occurred on ICS networks [41] and some reports state that data deletion attacks were executed against Iran's Oil Ministry control systems in Kharg Island around the same timeframe as the Saudi Aramco sabotage [42]. Shamoon executes a copy of itself as a scheduled job, entrenches itself as a service, drops a malicious driver which is loaded and executed to obtain disk access, then overwrites disk data starting with user data then system data and eventually the system's master boot record (MBR). The cyber attack effectively wiped 30,000 computers from Aramco, disabled some of its internal networks for weeks, and was soon spread to other oil and gas firms, such as RasGas [43].

In 2014, the first publicly-released prototype PLC rootkit with multiple payloads that can remain embedded in the device’s firmware appeared [44]. The rootkit conducts denial of service attacks and overwrites flash memory to make the attack persistent through device reboots. Planned future efforts by the researchers include the ability to distribute the malicious rootkit to other PLCs using available communication channels and adding telemetry data tampering functionality to the rootkit [44].

2. Commodity Malware in the ICS Domain

Malicious activity in the ICS domain can include the intentional or accidental use or re-purposing of known malicious software such as crimeware or banking Trojans. Even though these exploits and malicious software were not written to target ICS systems, commodity malware from traditional systems introduces instability into an already volatile ICS domain, with unknown effects on the operation cyber-physical systems. Because antivirus signatures exist for much broad-audience malware, the detection rate is high and there is more publicly available technical information than for the other categories. Often, regulated critical infrastructure entities must report these incidents and thus more are disclosed. We describe some representative examples.

In January 2003, the Nuclear Regulatory Commission (NRC) confirmed that the Microsoft Structured Query Language (SQL) Server worm known as the Slammer worm infected the Davis-Besse nuclear power plant in Oak Harbor, Ohio. The worm spread indiscriminately and infected one of the plant’s contractor computers, which consequently bridged a fiber optic connection from their computer to the internal SCADA network of the plant [45], bypassing a firewall that was configured to block the specific user datagram protocol (UDP) port used for propagation. The worm infected an unpatched server in the ICS network and caused enough network congestion to shut down the Safety Parameter Display System for nearly five hours. Reporting indicates that the Slammer worm also impacted other electricity sector systems [45].

In August 2003, a variant of the Sobig worm was introduced into the CSX Railway headquarters in Jacksonville, Florida. This commodity malware installed applications and created backdoors while continuing to spread by infecting e-mail

attachments. Although not specifically designed for or targeting the railway systems, the worm propagated into the control center causing loss of signaling, dispatch, and other related systems. Reporting indicates that Amtrak trains in the area were also affected by the malware's introduction into CSX Railway's systems. The end result was multiple train delays and expensive clean-up costs.

In August 2005, thirteen United States automobile manufacturing plants operated by Daimler Chrysler were shut down when the Zotob worm was introduced into the control network. This commodity malware was introduced to the ICS environment despite the existence of professionally-installed firewalls and studies of the incident concluded that a secondary pathway bridged network zones [46]. The plants' ICS networks remained offline for almost an hour and additional plants in Illinois, Indiana, Wisconsin, Ohio, Delaware, and Michigan were also forced down. The outage forced approximately 50,000 assembly line workers to cease work and cost Daimler Chrysler an estimated \$14 million in losses [46].

In October 2006, an attacker penetrated the network at a Harrisburg, Pennsylvania, water filtering facility [47]. The hacker compromised an employee's business laptop over the Internet and then introduced malware into the water facility's SCADA system using the compromised laptop's trusted remote access mechanism. The intrusion was discovered prior to damage occurring and thus the systems remained stable and operating.

In August 2008, viruses intended to steal passwords and send them to a remote server jumped a significant physical air gap and infected laptops inside the International Space Station. Although the impacts were minimal, the virus did make it onto more than one laptop, suggesting that was spread using internal networking or portable media.

In early 2012, ICS-CERT provided on-site support at a power generation facility where common malware had been discovered in the ICS environment and was likely introduced accidentally by an employee's use of a USB drive to back up control configurations. Several machines were likely affected by the incident, including two Windows-based engineering workstations, both critical to the operation of the control

environment. According to ICS-CERT, “these workstations had no backups and an ineffective or failed cleanup would have significantly impaired their operations” [48].

In October 2012, ICS-CERT responded to an incident in which the Mariposa scamming botnet was discovered on ten computers within a power company’s turbine-control system. This incident was caused when a technician used a USB drive to upload software updates during a scheduled equipment upgrade outage. Quarantine and restoration procedures at the company resulted in downtime for the impacted systems and a delay of the plant restart for three weeks [48].

In January 2014, malicious actors modified the widely-available “Gh0st RAT” Trojan and infected the Monju fast-breed nuclear reactor in Tsuruga, Fukui Prefecture [49]. RAT stands for remote administration tool and also remote access Trojan. Commodity RATs like Gh0st RAT and Shady RAT have been used for intrusions into critical infrastructure networks since at least 2009 [42]. Using a webshell hidden within an image on a popular media player’s update server, Gh0st RAT was downloaded to one of the eight computers in the Monju reactor’s control room [50]. Analysis of the incident has revealed that the webshell had been in place since 2011 and the modified Gh0st RAT malware was compiled in 2013, indicating that the adversary was patient and planned out the intrusion. The webshell redirected users at the Monju plant to another web server where a self-extracting compressed archive that contained multiple malicious modules was downloaded and various persistence mechanisms were established. Outbound command and control traffic was eventually manually observed by on-duty personnel while filing paperwork on the system, weeks after initial infection. The harvesting module of this malware allowed the compromised machine to be accessed more than thirty times in a five-day period and resulted in the pilfering of over 42,000 e-mails, meeting materials, Monju re-organization documents, and staff training records stored on the machine [49]. This intrusion only added to the troubled history of the Monju nuclear reactor and, following the intrusion, the plant was selected for decommissioning.

3. Unauthorized Persistent ICS Network Access

A third category of ICS incident refers to any unauthorized persistent access on control center systems or field devices from another network such as from the corporate network or the Internet. This is generally accomplished by subverting access controls for built-in remote ICS network access solutions, for which malicious use is difficult to detect. Interestingly, this category is closest to the official DHS definition of an ICS threat, which is “persons who attempt unauthorized access to a control system device and/or network using a data communications pathway, either trusted internal users or remote exploitation by persons unknown via the Internet” [51]. In fact, between 2001 and 2006, 70% of security incidents involving ICS networks originated outside of the network [52].

In 1992, a fired employee hacked into Chevron’s systems in New York and San José, California, reconfiguring the emergency alert network so that it would crash [53]. The intrusion was not discovered until an emergency occurred at a Chevron refinery in Richmond, California and the system could not be used to notify the adjacent community of the release of a noxious substance. Network configuration details have not been made publicly available to understand if and how the emergency alert network was interconnected with ICS devices, but the critical alert system was intended to cover twenty-two states and several regions of Canada. The unauthorized tampering resulted in a ten-hour system outage [53].

In March 1997, a Boston teenager connected his personal computer without authorization to a dial-up loop carrier system servicing the Worcester airport and subsequently sent a series of commands that disabled it [54]. The teenager’s actions altered the integrity of the data and severed communication links to an airport, forcing air traffic controllers to rely on manual overrides and backup systems for six hours. Additionally, the unauthorized access resulted in the disabling of phone service to over 600 homes in the area and affected the local weather service and fire department. Most notably, the teenager’s access temporarily disabled the radio transmitter that allowed aircraft to send an electronic signal to activate runway lights on approach. Information from the criminal case indicates that the loop carrier system operated by the telephone

company was accessible via modem to allow technicians to quickly change and repair service to customers from remote computers.

In April 2000, Vitek Boden wirelessly operated sewage treatment equipment at the Maroochy Waste Water facility in Queensland, Australia using a laptop in his car [2]. Over the course of three months, Vitek Boden successfully intruded into the SCADA system 46 times, falsifying his network address, sending false data and instructions, and disabling alarms that released thousands of gallons of raw sewage into rivers, parks, and tourist destinations [2]. The series of unauthorized accesses resulted in residents relocating due to the foul smell, the discoloration of the local waterways, over \$1 million in estimated costs, and substantial loss of marine life [3]. During the subsequent investigation, it was discovered that Boden previously worked as an engineer at the vendor supplying Maroochy's ICS components and he was attempting to obtain a consulting job to solve the problems he was creating [2].

In August 2006, two Los Angeles city employees compromised the network that controlled the city's traffic light system. [55]. The pair disrupted traffic signals on the signal control boxes at four critical intersections, resulting in significant backups and delays. This particular intrusion was launched prior to a labor protest by city employees.

In 2009, a 28-year-old disgruntled former IT contractor for Pacific Energy Resources remotely hacked into computerized controls that detected leaks on off-shore oil platforms off the coast of California [56]. Mario Azar's unauthorized access resulted in the crash of telemetry systems and operational data unavailability. The systems were disabled for almost two months before the intrusion was identified. This ICS safety-component failure caused thousands of dollars of damages for Pacific Energy Resources but did not trigger any leaks or physical consequences.

In November 2011, a hacker who called himself "Pr0f" demonstrated that by using the Internet-connected device enumeration tool Shodan, he could access HMIs within a South Houston water utility's ICS network [5]. Pr0f found that the water utility was running the Windows-based Siemens Simatic HMI software, a web-based dashboard for remote access to their SCADA systems, and was connected directly to the Internet

with only a simple three-character password for protection. According to ICS-CERT, this is one of countless successful unauthorized ICS network intrusions as a result of the exponential growth of Internet-connected ICS systems [48].

In May 2013, United States intelligence agencies traced an intrusion into the United States Army Corps of Engineers (USACE) National Inventory of Dams (NID) to a suspicious IP address [57]. The database contained sensitive details on vulnerabilities of the 8,100 major dams across the United States. Although technical details are not available to determine the interconnection between this database and the ICS network, the malicious user did maintain persistent access to the NID database for at least three months [57].

In May 2014, ICS-CERT reported that a sophisticated threat actor accessed a company's SCADA server that operated mechanical equipment. The SCADA server was directly connected to the Internet via a cellular data connection with no firewall or authentication controls in place. The intruder maintained access to the system over an extended period of time and connected over hypertext transfer protocol (HTTP) and SCADA-specific protocols, although no man-in-the-middle injection or system manipulation was observed [26]. In the same May 2014 release, ICS-CERT also disclosed that a public utility was compromised by a malicious group that gained unauthorized access to the utility's control systems. The ICS network was configured for remote access capabilities over the Internet with password-based authentication which the attackers were able to crack with brute-forcing [26]. The public release did not include any additional technical data or statements regarding impacts to the utility's operations, but the ICS-CERT report notes that the system owners were previously unaware of the insecure configuration and that the targeted systems were likely subject to prior intrusion activity.

B. KEY POINTS AND TRENDS FROM ICS INCIDENTS

The studied intrusions often started within the business network, or the business network was compromised as a reconnaissance point for follow-up intrusions into the ICS networks. By compromising the underlying operating systems of the workstations

hosting ICS software, adversaries were able to abuse trust relationships between those compromised systems' software applications and ICS hardware. Any action available to an ICS system's legitimate root-level Linux user or more commonly a system-level Windows user may also be performed by an adversary without the use of any ICS-specific knowledge or exploit tools.

In all studied incidents involving intentional malicious impacts to ICS systems, the adversary maintained a presence on the target system and had a communication method to communicate orders. In rare cases, malicious access was achieved directly to the field devices over radio or other connection method to the endpoints, however the majority of incidents involved the compromise of Windows-based supervisory workstations that monitor and control field devices. Targeted attacks and non-targeted commodity malware events also incorporated early rogue connection to the ICS network and thus unauthorized access is a prerequisite for all incident categories. Focused efforts to identify attempted or achieved unauthorized ICS network access may provide valuable early indication of many types of malicious activity.

Because availability requirements often prohibit the restarting of supervisory ICS machines, less sophisticated methods were required to maintain access to the target networks. Unauthorized access generally persisted for a significant amount of time in these case studies allowing attack planning and reconnaissance. The relatively simple persistent access methods and lengthy undetected malicious access to ICS networks were not reliably identified, according to several post-incident analyses, in part due to the constraints of auditing these systems and concerns of introducing instability into the environment.

Technical indicators of anomalous access and ICS network compromise have been extracted from these incidents' publicly available technical reports. The incidents used malicious methods to target ICS field devices as well as more traditional intrusion hacking techniques like establishing external command and control, scheduling tasks and modifying the registry to survive reboots, tampering with processes and services, implanting files for future action, laterally moving within the network, exploiting portable media devices, and abusing trusted channels to obtain and maintain an attack

position on critical-infrastructure systems. The spectrum of methods was similar to that of attacks on computer systems and networks in general therefore existing effective identification techniques should be carefully adapted for use in ICS environments.

IV. METHODOLOGY

A. FRAMEWORK FOR IDENTIFYING MALICIOUS ACTIVITY

The design of a technique to identify malicious ICS activity requires careful planning to obtain unique ICS data sources and collect and analyze that data under the strict operational constraints of critical networks.

1. Framework Overview

The proposed malicious activity identification methodology consists of collection, analysis, and decision components for host-based and network-based ICS artifacts (Figure 3). The framework is modeled after modern intrusion detection techniques employed in traditional networks [13]; however, these solutions are rarely deployed correctly for ICS networks at critical infrastructure sites [15] and most lack ICS protocol support as well as the signatures and behavioral-anomaly data necessary to identify ICS attacker tactics [7]. The purpose of this technique is to quickly analyze an ICS environment's systems and their communications, attempt to interpret abnormalities with a minimal required baseline, and isolate possibly malicious activity and pathways on critical networks in support of security assessments and cyber incident response.

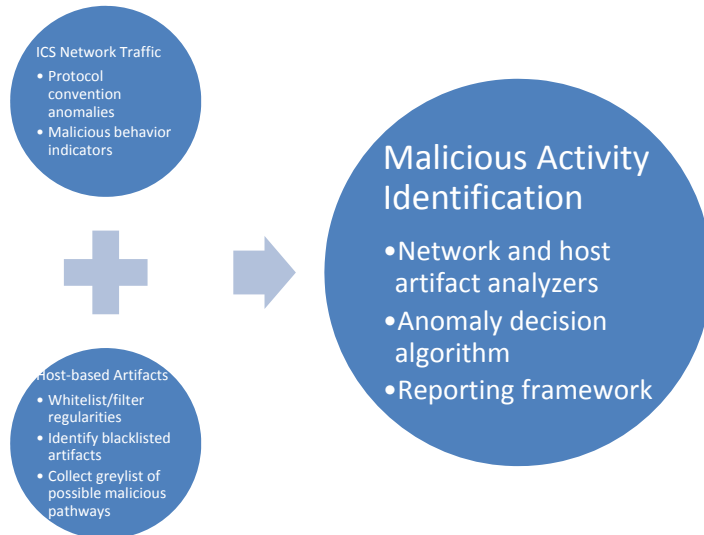


Figure 3. ICS Malicious Activity Identification Methodology

Although the goal is the creation of a toolkit that executes this methodology with reliable high-confidence output, the approach may need to be conducted manually at first and continually refined with automation in mind.

2. Framework Constraints

ICS networks require that specific technical constraints be understood and adhered to for both host-based and network-based tools.

Host-based tools are limited to those installed and already available on legacy operating systems. There exists no centralized view of system security status to draw data from, and field devices do not always store logs. Any security-monitoring commands executed on these systems should be run at the lowest priority level to not interfere with critical processes. Furthermore, there are temporal challenges with ICS forensic data because process and state information is often overwritten at a rate that makes meaningful collection unviable or impossible [6]. The unique configuration and availability requirements of these systems that were explored in previous chapters necessitate the adaptation of all host-based techniques for compatibility and the thorough testing of all commands to ensure critical processes will not be interrupted.

Network-based techniques are constrained by the instability introduced by active tools in the environment. Port scanning and automated device interrogation techniques can crash ICS hardware by scanning too fast or by sending null or malformed packets. With modern SCADA services using internal Web servers, HTTP GET and POST requests can cause actual physical actions, so even Web-based automated tools should not be used actively on ICS networks. Industry best practices recommend, and a few examples in the next section highlight, the importance of remaining entirely passive for network traffic analysis. That analysis must also include the highly specialized and often proprietary ICS communication protocols. Although minimal “noise” (irrelevant traffic) should exist on ICS networks, data volume is an issue because the amount of real-time telemetry data and otherwise unrelated traffic can drown out possibly malicious packets. Furthermore, ICS domain interconnection methods and some regulator restrictions limit the ability to tap critical networks at multiple locations. For these experiments, passive

network traffic captures were examined that were collected using a high-capacity switch port analyzer (SPAN) port on a router or switch, a method that should be usable on any ICS network.

Several traditional penetration testing techniques can be translated for use on ICS networks as shown in Figure 4. For example, instead of running a network port scan that sends packets intended to elicit device information that could trigger unexpected results, a penetration tester should verify open ports on each host in the environment without generating network traffic. This thesis explores eliciting considerably more detailed incident response and forensic data from these systems while following similar constraints.

Activity	Usual Actions for IT	Preferred Actions for SCADA
Identification of hosts, nodes, and networks	Ping Sweep (e.g. nmap)	<ol style="list-style-type: none"> 1. Examine CAM tables on switches. 2. Examine router config files or route tables. 3. Physical verification (chasing wires). 4. Passive listening or IDS (e.g. snort) on network.
Identification of services	Port Scan (e.g. nmap)	<ol style="list-style-type: none"> 1. Local port verification (e.g. netstat). 2. Port scan of a duplicate, development, or test system.
Identification of vulnerabilities within a service	Vulnerability Scan (e.g. nessus, ISS, etc...)	<ol style="list-style-type: none"> 1. Local banner grabbing with version lookup in CVE. 2. Scan of duplicate, development, or test system.

Figure 4. Comparison of IT and ICS Penetration Testing Actions, from [58]

Accounts exist of security researchers ignoring these constraints. For example, a ping sweep was performed on an active ICS network that controlled 9-foot robotic arms and a controller for an arm that was in standby mode received the ping sweep and abruptly swung around 180 degrees, luckily missing the person nearby [58]. In another instance, a ping sweep was performed to enumerate all hosts on an ICS network and it

caused an integrated circuit fabrication system to fail, destroying wafers worth \$50,000 [58]. In a third instance, a penetration test at a gas utility locked up the SCADA system and the utility was unable to send gas through its pipelines, causing a loss of service to customers, for four hours [58]. Lastly, in August 2006, operators at Browns Ferry Nuclear plant had misconfigured products from two different vendors, which resulted in excessive traffic on the control network. This excessive network traffic resulted in a high-power, low-flow condition where the recirculating water system could not be properly cooled [59]. This event, which took the plant offline for two days and cost nearly \$600,000 in revenue [59], demonstrated the fragility of these networks when exposed to unexpectedly heavy network traffic. These examples are proof of the complex requirements of any security assessment or forensic action on conducted on production ICS networks.

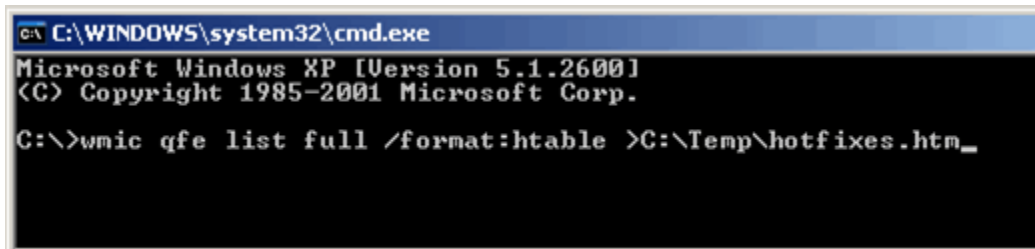
B. TOOLKIT SELECTION

An ideal toolkit requires the hand-selection of the most compatible host- and network-based tools capable of collecting and analyzing malicious activity while still providing ample ICS network coverage.

1. Host-based Tools

To aid stability of hosts, an implementation should focus on agentless built-in commands that generate minimal network traffic. Popular toolkits such as Microsoft's Sysinternals require detailed configuration and have not been thoroughly-tested on ICS systems, so they are not ideal for widely-applicable solutions. For host-based querying for Windows artifacts, only built-in command-line utilities should be used, with an emphasis on the Windows Management Instrumentation Command-line (WMIC) tool. Special care should be taken to ensure that the toolkit only attempts to query using command-line utilities that exist on that hardware's possibly-modified operating system. When possible, scripts should be written for the legacy Microsoft Disk Operating System (MS-DOS) 16-bit command.com processor that preceded the 32- and 64-bit cmd.exe found on Windows.

The implementation of these scripts will be compliant with North American Electric Reliability Corporation (NERC) Critical Infrastructure Protection (CIP) standards' scanning policies [60], which ensures that the toolkit can be used at not only electrical utilities but at other critical infrastructure sites with ICS systems that are subjected to the same scanning limitations. The tools are available on several versions of Windows operating systems, including most of those found within ICS environments, as illustrated within Siemens Automation documentation for the use of WMIC to troubleshoot system patching (Figure 5), representing a small subset of this thesis' planned usage. "Product work-around" documentation from General Electric (GE) shows that WMIC is also supported on a range of GE's platforms including GE's Proficy data historian platform and GE's Cimplicity HMI platform [61]. References to WMIC can be found buried within most major manufacturer's supervisory ICS system's troubleshooting documentation. While no documentation exists on the use of these tools for forensics or live response on control systems, tailoring ICS-compliant queries can identify unauthorized access based on real-world malicious ICS activity.

A screenshot of a Windows XP command prompt window. The title bar at the top reads "C:\WINDOWS\system32\cmd.exe". The window content shows the following text: "Microsoft Windows XP [Version 5.1.2600] Copyright 1985-2001 Microsoft Corp." followed by a command prompt "C: \>". The command entered is "wmic qfe list full /format:htable >C:\Temp\hotfixes.htm_".

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
Copyright 1985-2001 Microsoft Corp.
C: \>wmic qfe list full /format:htable >C:\Temp\hotfixes.htm_
```

Figure 5. WMIC usage on Siemens SIMATIC WinCC HMI, from [62]

These tools can run at user privileges but offer the most functionality when run as an administrator. For the research and testing purposes, a new shared administrator account with strong authentication was created on every test workstation to centrally query the hosts. Documented best practices recommend banner grabbing as a safe activity [58] and this WMIC node querying method uses similarly minimal network bandwidth. For critical real-world networks, the toolkit can run the commands locally with no network traffic with the manual export of results to be collated on a separate closed network.

2. Network-based Tools

The experiments used the Bro platform for network-based analysis [75]. It is an open-source network solution composed of signature detection, anomaly detection, and a programming language designed to work with network traffic. The signature detection generates logs for which a protocol analyzer abstracts details in real-time. Alerts can be generated based on pre-configured signature and anomaly notification rules. The programming language defines the actions the platform takes based on logic and structured programming. Fortuitously, the Bro network programming language was updated in November 2013 to support protocol parsing of the two most popular ICS protocols, Modbus and DNP3. Bro can parse and analyze network traffic and analysis can be automated through the creation of customized scripts to identify malicious ICS network activity.

Figure 6 lists the default protocol fields parsed by Bro for Modbus and DNP3. Bro's ability to extract function codes from ICS protocol messages can prove valuable in comparing network traffic against ICS protocol conventions. The Modbus and DNP3 analyzers process significantly more data than that provided in Figure 6, so custom scripts can be written to manipulate register addresses, values, and additional payload data. The created toolkit should scale in function as future Bro ICS protocol analyzers are developed.

ICS Protocol Data	Data Type	Modbus Field	DNP3 Field
Message timestamp	<i>time</i>	ts	ts
Connection unique ID	<i>string</i>	uid	uid
Connection orig&resp host&port	<i>record</i>	id	id
Message function code	<i>string</i>		fc_request fc_reply
Message failure exception code	<i>string</i>	exception	
Response internal identification #	<i>count</i>		iin

Figure 6. Modbus and DNP3 Bro log fields

Additionally, several open-source programming libraries such as jQuery and jVectorMap can be used to aid in reporting and visual effects for the network-based analysis toolkit.

3. Toolkit Coverage

Because the most skilled adversaries strive to maintain a minimal footprint, full inspection of ICS host and network elements is desirable but time and processing constraints, as well as the devices' operational limitations, make it infeasible at this time. Host-based and network-based tools can provide ample coverage of most ICS networks, while respecting constraints of device operation and real-time availability. Sufficient host-based sampling has been achieved and full population network-based data can be processed with this method, limited only by the ICS network operator's ability to collect and store all traffic.

Host-based tools should be selected to provide very high data-driven, non-probability judgment and convenience sampling rates. Host-based tools should be used locally on compatible hosts with minimal network traffic generated. Special logic should be included in the host-based scripts to ensure the stability and compatibility with various legacy versions of the Microsoft Windows OS. When possible, all scripts should be generated with MS-DOS and Windows 4.x commands, maximizing ICS host coverage.

The network-based analysis should be completely passive by design to not interfere with reliable process control. Analysis should cover the spectrum of systems communicating over TCP, UDP, and Internet control message protocol (ICMP) protocols on the ICS network, regardless of operating system. Specifically, the network toolkit should cover traffic for all hardware including field devices that communicate over Modbus TCP and DNP3, widely considered the two most-implemented ICS protocols in industry [18]. Support for more ICS network protocols can be extended as time permits.

C. IDENTIFICATION OF ADVERSARY TACTICS

This section details proposed technical identification techniques for adversary tactics as observed in the malicious ICS activity case studies. When applicable, host and network script excerpts are provided to illustrate usage.

1. ICS Field Device Anomalous Operations

In several cases examined, the only indication of malicious access was the anomalous operation of the ICS field devices. Stuxnet resided on the WinCC HMI and, using a known hardcoded password, modified rotating motor spinning frequencies in Siemens S7-315 programmable logic controllers and valve settings in S7-417s. This suggests that cleartext protocol data should be examined and ICS protocol datagrams should be inspected for known default passwords and other vulnerabilities. Obtaining and validating upper and lower field device register-boundary values against site-specific expectations and equipment-tolerance values may help identify overclocked or maliciously manipulated devices. ICS protocol operations should also be used to automatically create a catalog of devices, such as Modbus function code 0x11 and 0x2B that query for device details; responses to these packets can be passively inspected for field device characteristics. To extract high and low values, the maximum and minimum values for each device register should be stored and checked upon single register request, single register response, and multiple register request events. One of this study's Bro scripts can access and catalog these register-boundary values from both historical and real-time network traffic. The script should capture register-boundary values from replayed historical samples then create expected register limits with which to monitor real-time traffic, which requires fewer calculations and events. Because the ICS network protocol data is passively ingested on network infrastructure SPAN ports, and Bro is designed to distribute and manage loads [75], no network latency will be introduced when running any network-based scripts on historical or real-time traffic. If too much network traffic is aggregated, packets will merely be dropped on the monitored SPAN port; both ICS process control and network infrastructure operations will continue without degraded or disrupted performance.

The analysis of ICS field device register-value ranges requires additional information such as the make and model of equipment and what process it may be controlling. Organizationally unique identifier (OUI) bits should be extracted from media access control (MAC) addresses to assist in fingerprinting devices for both overall awareness and for verification of specific ICS hardware limits. A repository of ICS-specific OUI vendor information and possible field device function has been researched and created for this thesis for offline component identification. Unfortunately, Bro abstracts network traffic at an early phase of analysis and does not currently allow the extraction of MAC addresses from ICS protocol communication.

To assist, dynamic host configuration protocol (DHCP) traffic within the ICS network traffic should be examined for physical addresses, modifying an existing Bro policy and further extending it with ARP, DNP3, and Modbus protocol analysis. Additionally, the host-based scripts should query each system to extract MAC addresses from their local ARP tables for devices with which they have communicated, and from any in-range wireless infrastructure (Figure 7). This method should assist in monitoring for ARP poisoning as well. Dual-homed devices, those systems that have multiple network interface cards, are attractive pivoting targets for malicious attackers, so any MAC address with multiple IP addresses should also be identified.

```

echo MAC Address Vendors from IPCONFIG: && echo -----
FOR /F "tokens=11 delims=: " %A IN ('IPCONFIG /ALL ^| FINDSTR /R
"[^-][0-9a-fA-F][0-9a-fA-F]-" ^| FINDSTR /V "Tunnel" ^| FINDSTR /V
"00-00-00-00-00-00") DO (
    set MACAddress=%A
    set Vendor=!MACAddress:~0,8!
    FINDSTR /I !Vendor! vendorMacs.txt
)
echo.
echo MAC Address Vendors from ARP TABLE: && echo -----
FOR /F "tokens=2" %A IN ('ARP -a ^| FINDSTR /R "[^-][0-9a-fA-F][0-9a-fA-F]-"') DO (
    set MACAddress=%A
    set Vendor=!MACAddress:~0,8!
    FINDSTR /I !Vendor! vendorMacs.txt
)
echo.
echo MAC Addresses for Nearby Wireless: && echo -----
FOR /F "tokens=4" %A IN ('NETSH wlan show networks mode^=bssid ^| FINDSTR /R
"[^-][0-9a-fA-F][0-9a-fA-F]:"') DO (
    set MACAddress=%A
    REM Reformat for offline search database xx:xx:xx to xx-xx-xx
    set Vendor=!MACAddress:~0,2!-!MACAddress:~3,2!-!MACAddress:~6,2!
    FINDSTR /I !Vendor! vendorMacs.txt
)

```

Figure 7. Host-based passive ICS device enumeration batch script

Since ICS traffic often should be deterministic (the operation of a device should be predictable since critical processes rely on predictable outputs for given inputs) the network traffic can be compared against ICS traffic conventions to isolate tampering or anomalous behavior. This provides a stronger indication of malicious activity than on traditional computer systems. One such convention is the hierarchical communication where devices communicate one-to-many or master-to-slave. Master and slave roles can be easily extracted for IP-based protocols because the source and destination addresses for a specific message type imply its role. Communication channels are defined by device function so an HMI should only talk to a PLC, RTU or SCADA I/O server and an RTU should probably not be sent non-ICS protocol traffic; these should be constant because devices are rarely added or removed from the network. Another convention is that ICS field device communication is consistent and so, where on traditional IT networks the packet timing is influenced by human interaction, the ICS protocol traffic generally occurs on set polling intervals.

Additional logic should be added into the Bro scripts for ICS field devices to check for suspicious functions and operations that require an explicit interactive command or reprogramming. Engineers and operators, when presented with this data, would know instantly if it was a command that they issued or if it was a possible unauthorized injected command. The Bro script should include specific verbiage detailing the potential security concern of the ICS field device command transmitted. For Modbus, function code 0x7D is issued to initiate firmware replacement. DNP3 code 0x1B is issued to delete a file and codes 0x0D and 0x0E are issued to restart a device. Those commands would be instantly verifiable if detected in live ICS traffic. Less critical alerts should also be included in the scripts but require operator analysis to determine if they are malicious or if the ICS network is misconfigured. For instance, a Modbus slave's exception code of 0x03 means an illegal data value was received. Since we have established that Modbus is hardware-agnostic and not aware of device register limits, this exception means that the structure of a query was unexpected. Modbus exception code 0x0B indicates that a target device failed to respond or may not be present on the network. DNP3 uses code 0x21 for an authentication error and 0x82 for an unsolicited response, both of which should be extracted by the created scripts, but it will require an operator to discern whether the presence of the codes indicates malicious activity or device misconfigurations.

Comparing observed ICS communication against protocol conventions like these should allow the identification of malicious persistence without the need for an anomaly-based learning period.

2. External Network Communication

The majority of cases studied included some form of malicious communication to external networks. The high signal-to-noise ratio within IP-based ICS networks should make the identification of malicious external communication significantly easier when compared to traditional enterprise networks.

All versions of the Mariposa malware use custom UDP datagrams for communication, and infected systems may beacon frequently and send encrypted

instructions and data across a variety of UDP ports. Hardcoded domain names are used in most variants as well. The Monju incident's malware redirected from the compromised streaming media service's website to another malicious domain and used command and control traffic over TCP port 443. The connections were established with malicious server testqead.tk, so domain name system (DNS) lookups over TCP port 53 would be observed [50]. DNS cache, cookies, and the hosts file can be examined for previous outbound attempts and may help identify planned routes or dormant malware. Web browsers also store typed uniform resource locators (URLs) in various forensic locations, such as in HKCU\SOFTWARE\Microsoft\Internet Explorer\TypedURLs for Internet Explorer.

Since this thesis has argued that ICS networks should not connect to the Internet directly, all attempted or successful connections to external IP space may be of concern, unlike business networks. For instance, both a half-open TCP handshake and a connection that has been rejected by an external IP issuing a reset packet to deny it would not be considered a connection in most traditional networks. Any external connection attempts using ICS protocols are of major concern and they should be prioritized for the user of the toolkit to review.

Figure 8 provides an excerpt from this thesis' Bro script to categorize potentially malicious connections outside of the ICS network. The first function abstracts and simplifies the state of every TCP and UDP connection based on the status of its three-way TCP handshake. Selecting an event-based trigger that captures both attempted and successful connections in network traffic can be difficult; however, the second function shown in Figure 8 should trigger when a connection's internal state is about to be removed from memory. Every TCP and UDP unique 4-tuple socket pair should reliably generate an event at which time the network-based toolkit should apply logic to filter out the original, meaningful external connections and their connection statuses. The bottom of Figure 8 also displays a portion of the robust ICS port and protocol pairing that should be matched to newly-identified external connections. This ICS protocol tagging should provide helpful context for determining the security risk of a particular connection since a

previously-unknown connection may be of more immediate concern if it is communicating over an ICS protocol.

```
function icsState(intState: count, extState:count, trans: port): string {  
    if ( get_port_transport_proto(trans) == tcp) {  
        if (extState != TCP_INACTIVE) {  
            if (intState == TCP_ESTABLISHED || intState == TCP_CLOSED) return "ESTABLISHED";  
            else return "PATH EXISTS";  
        } else if (intState != TCP_INACTIVE) return "ATTEMPTED";  
    } else if ( get_port_transport_proto(trans) == udp) {  
        if (extState == UDP_ACTIVE ) {  
            if (intState == UDP_ACTIVE) return "ESTABLISHED";  
            else return "PATH EXISTS";  
        } else if (intState == UDP_ACTIVE) return "ATTEMPTED";  
    } else return "OTHER"; # TODO: Add ICMP connection state logic  
}  
  
event connection_state_remove(c: connection) {  
    local respIP: addr = c$id$resp_h;  
    if ( is_v4_addr(c$id$resp_h) ) {  
        if ( c$id$resp_h !in reservedprivate && c$id$resp_h !in reservedlocal && c$id$  
resp_h !in reservedcast ) { # subnet sets based on IANA RFC 1700, 1918, 6890  
            local loc_resp = lookup_location(c$id$resp_h);  
            if (icsState != "OTHER") {  
                writeLog(c$id, ics_protocols[c$id$resp_p], lookup_location(c$id$resp_h),  
icsState(c$orig$state, c$resp$state, c$id$resp_p));  
                addBroMAPnode(); # format jVectorMap marker output, JSON::convert() strings  
            }  
        }  
    }  
}  
  
redef ics_protocols = { # Externally communicating protocols of interest  
    [102/tcp]   = "ICCP",  
    [502/tcp]   = "Modbus_TCP",  
    [1089/tcp]  = "Fieldbus",  
    [1089/udp]  = "Fieldbus",  
    ...
```

Figure 8. External ICS network communication identification Bro script excerpt

Passive IP address geolocation can be conducted within this same script using the offline MaxMind GeoIPLite database to plot attempted and established connections from the ICS network and the protocol used. The geolocation and plotting of external connections on a map should help to determine approximately where anomalous traffic is bound. This method would assist in distinguishing innocuous connections, like packets to several IP-enabled meters within a utility's service area, from nefarious connections, like recurrent unauthorized beaconing to a foreign country's IP space over ICS protocols. Bro scripts can be written to convert and store relevant connection attempts using JavaScript object notation (JSON). Formatting processes can be conducted during Bro's initialization and completion to directly write output to a JSON file as shown in Figure 9.

This offline JSON data feed should be continuously-updated while also being projected onto a world map using the jVectorMap library.

```
var dataArray = [
/* [ { latLng: [lookup_location(c$id$resp_h$latitude), lookup_location(c$id$resp_h$longitude)] },
   name: JSON::convert(c$id$resp_h$city+", "+c$id$resp_h$region+", "+c$id$resp_h$country_code),
   ics_protocols[c$id$resp_p], c$id$orig_h, c$id$resp_h,
   ICSconnectionState(c, get_port_transport_proto(c$id$resp_p)) ] */
[ { latLng: [38.778,-77.125], name: "Alexandria, VA, US" },
  "MODBUS_ICP", "192.168.1.227", "173.66.46.112",
  "ESTABLISHED" ],
[ { latLng: [37.304199, -122.094], name: "Cupertino, CA, US"},
  "HTTP/SSL", "192.168.1.13", "143.127.102.40",
  "ATTEMPTED" ]
```

Figure 9. Bro script variable translation to jVectorMap overlay data

Not only should external Internet access be disallowed but so should several other protocols within the ICS network and ICS DMZ [63]. FTP and trivial file transfer protocol (TFTP) are common ICS transfer protocols which use minimal processing power that is ideal for limited hardware [15]. Not only are FTP and TFTP vulnerable services, since cleartext passwords are used for FTP and no login is required for TFTP, but they are known to be used by malicious actors. Outbound FTP sessions could indicate malicious intellectual property theft, so despite the legitimate use of FTP in some ICS networks, any high-bandwidth FTP usage should be identified [64]. Similarly, inbound Telnet sessions from the corporate network, simple network management protocol (SNMP) versions 1 and 2, and any simple mail transfer protocol (SMTP) or other e-mail traffic should be flagged [65].

Shamoon's command and control (C2) traffic was configured to beacon home every two hours. The module responsible for sending information about the infection to the attacker would send an HTTP GET request including the domain name, number of files overwritten, IP address of the compromised system, and a random number [66]. If internal web servers are being used for SCADA services, examining irregular HTTP properties like user agent strings can help to identify automated attack tools or manually-performed attacks. Although the logging of requests is unlikely to be enabled for internal SCADA servers that use HTTP as a transport protocol, HTTP traffic should be extracted and characterized by the network-based toolkit.

Depending on the communication methods used between ICS devices, the enumeration of IEEE 802.1x wireless access points can detect rogue connections. NIST recommends that wireless access points be located on an isolated network with minimal connections to the ICS network [15]. Current in-range service set identifiers (SSIDs) should be enumerated on ICS hosts since not only has the existence of in-range SSIDs been used for previous non-ICS intrusions but in many critical environments there should be no wireless access points accessible. Additionally, data can be extracted using the “netsh wlan show profiles” command to indicate if a host has previously connected to any wireless access points. If run as an administrator, the same command can be modified with “key=clear” to list cleartext stored 802.11 wireless passwords, a technique that should be provided within this research’s scripts for administrator awareness.

3. Registry, Startup, and Scheduled Task Persistence

The Mariposa bot kit found on a turbine control network used a registry startup method that works on all Windows versions from all accounts, including limited guest users, by modifying the Winlogin registry entries to enable the bot to start at boot [67]. The primary malicious module of the Monju plant incident added entries to the Windows registry in one of two locations depending on Windows version and system architecture. Backdoor keys are placed in the default software classes under \InProcServer32\@=expand:”C:\WINDOWS\temp\install.ocx” within HKEY_USERS in the registry, allowing the malware package to launch on system startup [50]. Shamoon created the TrkSvr service that starts itself with Windows, which may have been identifiable by monitoring the drivers set to load during startup. Malicious services can be configured to automatically restart after specific service failures using the SC utility, a utility that can also be used to query services for this property.

Comparing the hosts’ registries against each other and also against a clean operating system baseline may be achievable in ICS networks, unlike most networks, since significantly less software should be installed and user behavior is restricted. In the absence of those registry baselines, several common locations are used for malicious registry persistence and several specific locations were used by the malware from the ICS

case studies. These locations, as well as the “shutdown” key value with a dynamic-link library (DLL) loaded, should be checked in the registry.

The Mariposa malware also overwrote the user’s Desktop.ini file that initializes on restart, a file often targeted for persistent access. Placing binaries and batch scripts within users’ startup folders is a simple but effective tactic for maintaining persistent access. In fact, during the Shamoon incident, attackers employed basic batch scripts to assist in data gathering. The relatively low number of users and overall workstations in the ICS environment should expedite the checking for these types of plaintext command-line instruction files.

Both Stuxnet and Shamoon used the task scheduler to create and then delete tasks, a process that should leave forensic artifacts for collection. Shamoon used the task scheduler to create At.jobs for privilege escalation to the system account and also to periodically run command-and-control modules. The task scheduler was ultimately used to execute the wiper modules (system32\dfrag.exe and system32\dvdquery.exe) as well. These names were selected from a hard-coded list of binary names to blend in with system files. The tasks for Shamoon included logic for execution times, with the Triggers/TimeTrigger property of the scheduled task as well as Actions/Exec for the explicit command execution. If a high number of legitimate scheduled tasks exist in the environment, only those with logic triggers or command-line execution need to be examined for suspicious behavior.

Technical incident data can reveal multiple methods of scheduling Windows tasks for both persistence and privilege escalation. It is necessary to check tasks scheduled using a variety of services, such as AT jobs, BITSADMIN jobs, and jobs assigned with SCHEDULE_TASKS. To further separate potentially malicious tasks from innocuous administrative scheduled jobs, tasks with executable actions, explicit command line variables, and logic triggers such as time should be displayed prominently in the script’s output.

Scheduling a task on supervisory systems within ICS networks may also produce a log entry with EventID 602 for older versions of Windows or EventID 4698-4702 for

Windows 6.0 and up. Systems can also be configured to log Event ID 567 or 4657 when registry keys are modified so that a service is started at boot. Querying event logs can be processor-intensive and detailed logging is frequently unavailable on ICS hosts; however the scripts created for this thesis include relevant command-line search and extraction of available event log data.

4. Process Injection and Hijacking

Stuxnet replaced a DLL file used by Siemens Step 7 to communicate with PLCs, allowing Stuxnet to intercept, manipulate, and replay ICS traffic. While Stuxnet was sophisticated malware, cleaning the infection was as simple as disabling or deleting two signed driver entries, `mrxcas.sys` and `mrxnet.sys`, from the `Windows\System32\` directory [68]. These files, dropped early on in the infection, are registered to start when the system boots. Although it is difficult to check for changed file listings across all hosts in the business network, it may be feasible to monitor specific directories on ICS systems for malicious libraries intended for DLL search-order hijacking. Anomalous characteristics should be recorded for running processes and services, like the loading of non-Microsoft DLLs into `services.exe`, `lsass.exe`, or `explorer.exe` or processes launched from user and temp directories. Determining the running processes that are not executing from the operating system directory may be a valuable script for live production ICS systems.

An excerpt from this thesis' initial ICS running process attribute whitelist is shown in Figure 10. Using the command-line, WMIC queries should be structured to list all processes and services whose name or display name is not equal to those listed within an external lightweight file. Results from all hosts should be counted by unique process and output to a hypertext markup language (HTML)-formatted table.

Whitelist	Process	CCNSMozProvider.exe	Siemens	Simatic WinCC
Greylist	Port	445	Wonderware	IAS
Whitelist	Port	135	Wonderware	IAS
Whitelist	Port	1433	Wonderware	IAS
Whitelist	Port	1434	Wonderware	IAS
Whitelist	Process	CCPackageMgr.exe	Siemens	Simatic WinCC
Whitelist	Process	CCPerfMon.exe	Siemens	Simatic WinCC
Whitelist	Process	CCProfileServer.exe	Siemens	Simatic WinCC
Whitelist	Process	CCProjectMgr.exe	Siemens	Simatic WinCC
Whitelist	Process	CCPtmRTServer.exe	Siemens	Simatic WinCC
Whitelist	Process	CCRTsLoader.exe	Siemens	Simatic WinCC
Whitelist	Process	CCTextServer.exe	Siemens	Simatic WinCC
Whitelist	Process	CCTMTTimeSyncServer.exe	Siemens	Simatic WinCC
Whitelist	Process	CCTxtProxy.exe	Siemens	Simatic WinCC
Whitelist	Process	CCUCSurrogate.exe	Siemens	Simatic WinCC
Whitelist	Process	ChannelWrapperCS.exe	Siemens	Simatic WinCC
Whitelist	Process	DASABCI.exe	Wonderware	DA Servers
Whitelist	Process	DASABTCP.exe	Wonderware	DA Servers
Whitelist	Process	DASAgent.exe	Wonderware	DA Servers
Whitelist	Process	DASAlarm2U.exe	Wonderware	DA Servers
Greylist	File	%WINDIR%\System32\OPCEnum.exe	Wonderware	DA Servers
Whitelist	File	...	Wonderware	DA Servers

Figure 10. Excerpt from this thesis' ICS process and service attribute whitelist

The commodity malware used for the Monju reactor compromise used malicious implant code run by non-malicious trusted code, similar to DLL search-order hijacking with a signed executable [50]. Because the streaming media player service downloaded a malicious update, cleverly named as the setup for the beta version of the player, the malicious modules all ran at the same privilege as the legitimate streaming media player process.

Special attention should be paid to limit processes that are installed and listening on multiple boxes which, if compromised, provide access to multiple hosts [63]. With fewer systems and processes running than traditional IT networks, it should be possible to identify anomalous processes running in control centers using WMIC with the /node switch for remote hosts and a well-structured central query.

5. File System Sabotage

The malware used at the Monju plant hid itself with double file extensions, padding .dll files as .tmp and .pdf files [50]. Running a file-type classifier such as the

Linux “file” command or detecting extension mismatches in system or temp directories can help to identify previously-unknown malicious files. Not only can analysis be conducted on the hosts for these extension mismatches, but also within the network traffic captures.

While rare file extensions may be more common within ICS networks due to the systems’ specialized function and regular dependency on proprietary software, isolating non-standard software associations may help identify file types that have been registered to start with malicious programs. Additionally, the Windows registry debugger value can be set to hijack file associations to execute them with malicious software. This is accomplished by setting the debugger string value within the registry’s image file execution options for a given executable and can be achieved with a single command.

The Mariposa malware, a variant of which has been found on ICS networks, and several other well-known malicious programs copy binaries into the C:\RECYCLER folder. Specifically, Mariposa copies a file called dllrun32.exe, but finding any non-deleted files in the RECYCLER or newer \$Recycle.Bin directories may be of interest in assessing the security of the ICS domain. All files in C:\\$Recycle.Bin*(SID)* should have filenames that start with a ‘\$’. The recycling bin may also contain interesting recently-deleted data, so it should be checked for .exe, .rar., .zip, and .txt extensions to consider files for forensic recovery.

Although compressed archives may be common at some critical sites for packaging large amounts of data to move between segmented networks, the presence of anomalous compressed archives on ICS file systems has been linked to malicious activity and thus is included in this method. The logs of antivirus and other system scanning solutions may include historical information on encrypted compressed archives that could not be opened and inspected at a previous time, even if those archives no longer exist.

During the Monju reactor’s infection, a self-extracting compressed .rar archive was used to deliver components [50]. Shamoon used compressed archives on the network to package reconnaissance data including hostnames and IP addresses (hostnames.rar, ips.rar) and also to deliver malicious tools like Windows credential editor (wc.zip,

w64.zip). Even limited ICS hardware can run the WMIC command with path=cim_datafile and when passed a WMIC query language (WQL) “where” clause for the desired extensions should be able to list all compressed archives on Windows hosts in the environment.

Automated file listing and property checks can also include the auditing of certain applications like Windows Sticky Keys (sethc.exe) and Windows Utility Manager (utilman.exe) that are often replaced for privilege escalation and malicious persistence. These files generally require local access to modify but they are launched with specific keyboard sequences with administrator privileges. Another file listing trick that malicious users can exploit is the way Windows handles unquoted executables. If paths include spaces, such as ‘C:\Program Files\’ or ‘C:\Documents and Settings\,’ and references to these locations are missing the quotes around the full path, Windows will separate file items at the spaces. For the examples provided, C:\Program.exe and C:\Documents.exe would be executed if those files existed and references were missing fully-quoted paths. Auditing for unquoted executable usage can be conducted quickly with few false positives expected.

6. Internal Network Lateral Movement

Adversaries attempt to transition through internal networks to gain control or access on a system inside the target security zone from a network presence on a lesser security zone. Stuxnet accomplished this lateral movement by propagating over network shares, exploiting a printer-spool vulnerability, and through Windows remote procedure call (RPC) commands [39]. Stuxnet enumerated user accounts on the computer and attempted to explicitly login with the user’s credential token to all network resources and execute itself on remote shares. If Stuxnet determined that the ADMIN\$ share was accessible, it repackaged the malicious code with the latest configuration data block and copied itself to the share with a .tmp filename. A network job was then scheduled to execute the file two minutes later. Shamoon also copied itself to accessible network shares then, if successful, executed itself remotely. Network shares and shared printers

make attractive targets to span network zones so the toolkit should attempt to identify these pathways on ICS networks.

A module of Gh0st RAT used at the Monju plant allowed for peer-to-peer communications to further propagate through the target network [50], so processes listening or established internally on unexpected ports may identify malicious infrastructure.

Internal lateral movement can also appear as many flows from a single workstation to others that normally do not communicate, especially over ports 139 and 445. If logs are available, Event ID 552 in the Security event log may help identify accounts and systems used to conduct this activity, and these can then be filtered to isolate malicious behavior and lateral movement.

7. Portable Media Exploitation

The creators of Stuxnet probably used portable media devices for the initial infection vector, due to the Microsoft Windows shortcut ‘LNK/PIF’ automatic file execution vulnerability [39], allowing auto-execution from USB drives. Also, the 2012 power generation facility compromise and the turbine control network Mariposa infection published by ICS-CERT resulted from employee and contractor usage of infected USB drives [48]. Mariposa is known to use the AutoRun string to spread via USB [67]. ICS network hosts should be audited to ensure that AutoRun is disabled in the registry for portable media and unknown drives or devices. On Windows XP and below, this setting can be found in the Policies\Explorer\NoDriveTypeAutoRun keys within HKEY_LOCAL_MACHINE and HKEY_CURRENT_USER in the registry, so extraction and interpretation of these values from the command-line should be pursued.

Furthermore, most current operating systems and even some legacy systems provide queryable records of inserted portable media devices. Custom scripts can be created to check USB device connectivity, make and model of previous devices, and other metadata to determine if ICS network security policies are being followed and if malicious portable media exploitation has been performed. If the script is executed across multiple nodes using WMIC, statistical analysis can correlate unique identifying registry

values in HKLM\SYSTEM\CurrentControlSet\Enum\USBSTOR. Windows generates and stores a unique instance identifier, often the device's serial number, as a subkey in the registry and stores the InstallDate, LastArrivalDate, and LastRemovalDate as Property IDs 0x0064, 0x0066, and 0x0067 respectively. The container identifier can be cross-referenced with other variables elsewhere in the registry to determine if AutoRun was enabled and to find the specific drive label. This method should allow the ability to categorize portable media characteristics such as vendor and device model and timeline each unique device's usage across the ICS network.

8. User Activity Abnormalities and Remote Access Abuse

Malicious attackers often create user accounts or leverage stolen credentials for pre-existing accounts to conduct their desired actions. User accounts are difficult to manage on segmented networks with custom network architectures at each site for synchronizing active directory and the domain controller with the business network. It is expected that there may be some unused and incorrectly-configured accounts on ICS systems since user access may be more difficult to audit than on other systems. With comparatively few nodes on an ICS network, a full listing of user accounts can quickly and automatically be analyzed for various traits using command-line tools. Host-based artifacts, and the corresponding log events if available, should be analyzed for locked-out accounts, users who have never logged on, passwords that never expire, guest accounts belonging to a group, blank administration passwords, and the creation of new user accounts and their naming convention, and simultaneous sessions across multiple machines to discover possible compromised user accounts and intrusion pathways.

One specific behavior that a toolkit should attempt to identify is suspicious activity when users are not at work. Many of the studied incidents were either detected or later correlated with off-hour access into network. Leading up the Shamoon attack, persistence was established and surveillance was regularly conducted during off hours. This is a consistent trait for many sophisticated intellectual property theft intrusions on the business network as well, with over 97% of a sophisticated nation-state's 1,905 observed remote desktop protocol (RDP) infiltrations occurring during that country's

standard business hours, which are off-hours for the United States. For ICS networks, there may be a specific time window that any organization would not expect human interaction with process control. Although certain situations may necessitate after-hours remote access to an ICS network, these may be identified by plotting remote interactive logons (type 10) against time of day, and likely the anomalies could be explained by the staff.

Identifying user behavior anomalies has become more challenging with modern SCADA engineers' growing expectation of and reliance on remote access to ICS systems at other locations. The Worcester airport intrusion was conducted by abusing dial-up remote access intended for engineers. Since some ICS networks still use dial-up modems or wireless data for direct or secondary RTU access, these access points may provide valuable anomalies from normal engineer access. The Shamoon attackers enumerated RDP servers and attempted to use several legitimate tools, including PSEXEC, Net, and Mstsc to exploit trusted access pathways. ICS-CERT did not release which technologies specifically were used in the May 2014 sophisticated remote access abuse intrusion [26], however companion alerts issued reference RDP, virtual network computing (VNC), and Sysinternals tools including PsExec. To use any Sysinternals tool you must accept the EULA on that system which leaves a record within the registry, which presents a unique and quick way to determine these tools' presence from the command-line without searching the entire drive. To do this, the value should be checked for Software\Sysinternals\PsExec\EulaAccepted in the registry.

According to a document released by NERC, the Slammer worm that affected the Davis-Besse nuclear plant also downed another electric utility's critical SCADA network after moving from a corporate network, through a remote computer, to a VPN connection to the control center LAN [69]. VPN logs can be compared across hosts within the ICS network and across network zones. Information can be extracted from these and other related files to identify anomalous and possibly malicious usage.

Separating legitimate use of trusted remote access tools from malicious use is challenging, but several distinct properties of ICS networks may assist in developing a technique to reliably accomplish this. Despite being highly valuable for analysis, detailed

logging of remote access events in the ICS domain cannot be expected at all sites, and thus relevant forensic artifacts for these access methods must be collected from various locations in the file system using compatible command-line tools.

D. CONSOLIDATED TACTIC IDENTIFICATION

If an unauthorized user had previous access or currently maintains malicious persistence on an ICS network using any of the described adversary tactics, forensic artifacts should be available for careful collection and analysis.

Malicious tampering with ICS field devices communicating over industrial protocols may result in anomalous device operation and unconventional communication patterns. This activity should be identifiable with statistical analysis of function codes even without prior knowledge of the specific ICS site due to minimal network noise and field device behavioral expectations. Analysis should be conducted primarily on ICS network traffic with additional device functionality fingerprinting assistance from host-based queries.

Network traffic should be examined for attempted and established external connections, since client-side attacks require access from outside the ICS network. External connection and traditional protocol analysis should also reveal beaconing commodity malware as well as weak security practices that can or have been used as adversary access channels, with geolocation helping to triage previously-unknown connections. Host-based artifacts should be used to support historical external access.

Several known locations used for malicious persistence that survive system reboots, such as specific registry keys, system startup locations, and scheduled tasks should be examined for content that is unexpected on critical networks and also correlated among devices to identify software anomalies. Host processes and their imports should also be examined to ensure only expected minimal processes on critical nodes. Popular file system locations should be carefully examined to identify data-exfiltration staging areas and the modified files and extensions used for privilege escalation. The startup persistence, system process, and file system artifacts should be compared across hosts and analyzed with an ICS-specific attribute whitelist and blacklist.

For additional vulnerable ICS network access vectors, host-based artifacts should be extracted that provide context on internal network pathways, portable media usage, and remote access abuse. To increase analytical confidence in these findings, these forensic attributes should be checked with on-site ICS engineers and compared with network traffic to isolate malicious behavior from observed anomalies.

THIS PAGE INTENTIONALLY LEFT BLANK

V. EXPERIMENTS

A. DATASETS

A goal of this research was to design a reliable and effective ICS malicious activity identification toolkit. This chapter provides an analysis of the toolkit's operation and its investigative strengths and weaknesses. Tests were conducted on a variety of data sources, including voluntarily-submitted and publicly-available ICS network traffic as well as researcher-provided sophisticated advanced persistent threat (APT) malware and packet captures.

When sufficient technical data was unavailable on adversary tactics, techniques, and procedures (TTPs), actions were recreated on an array of virtual machines configured with a variety of operating systems and software representative of a typical ICS installment. The resulting forensic evidence was used to validate and enhance the host- and network-based scripts.

B. INITIAL RESULTS

The prototype toolkit consisted of eight substantial command-line utility scripts representing each adversary tactic researched, with several supporting batch scripts for adhering to ICS device limitations and formatting results. Although Bro is open-source software that includes many built-in functions and analyzers, several custom Bro network programming-language scripts were written that extend the coverage and capability of the methodology. I have authored all described host and network-forensic artifact-collection scripts for each specific adversary tactic explained. I have matured and improved the toolkit's scripts over time and several excerpts and iterations have demonstrated an ability to collect forensic details within the operational constraints of real-world sensitive environments at various critical infrastructure sites. The most recent versions of the file-sabotage, process injection and hijacking, and internal network lateral movement host-based scripts have not been tested on ICS networks but have performed as intended in the simulated environment.

The preliminary findings from testing the toolkit can be used to assess the methodology's performance and adherence to unique ICS constraints. The scripts will be continuously improved as more data is processed so that observed ICS activity and potential pathways can be directly translated into the toolkit's growing ICS forensic attribute whitelists and blacklists.

1. Operation within Constraints

To operate within the ICS domain without affecting ongoing operations, all host-based tools had to be run at the lowest priority level to allow process control functions to operate at full availability. This was accomplished by using the built-in Windows base priority function to force the process and thread priority to the lowest possible level [70]. Attempts to lower the host script's priority from within by referencing its own process ID (PID) produced inconsistent results. The most successful technique was to launch all host-based scripts from within a separate wrapper function that reliably set the priority for all spawned processes. Set at idle priority, the scripts only use free processing cycles and do not interrupt any functions. Idle priority is normally used for lightweight software such as desktop screensavers and applications that only require periodic updating. Within the master script, the user is given the option to increase this priority level if faster results are desired and the system can tolerate it. All host-based scripts were successfully launched at the lowest priority level and monitored to ensure that they adhered to this constraint (Figure 11). Additionally, disk reads and writes were minimized through similar process monitoring to safeguard against retrieving too much data, which could overwhelm a resource-constrained ICS system.

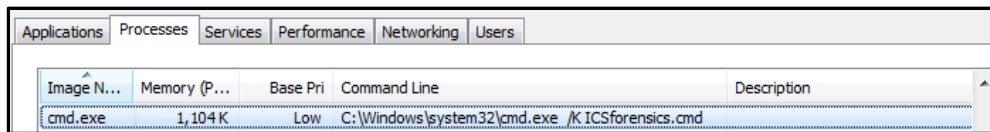


Figure 11. Host-based scripts running at lowest priority

The command-line utilities employed on the hosts were selected not only based on their ability to produce reliable results for identifying each adversary tactic, but also

for their support on legacy operating systems. The logic built within the scripts successfully refrained from executing on unsupported versions by using the VER command and checking the %OS% and %COMSPEC% environment variables. The WMIC command, which allows for detailed system management from the command-line, can be run with the “/node” option enabled to run the host scripts across multiple systems and centrally collect the data, while generating minimal network traffic.

The network-based toolkit successfully functioned without the need for active interaction with devices. This was confirmed by replaying historical network traffic through the scripts to produce the results, while running tcpdump to intercept and display packets on the system running the network-based scripts. Although not a requirement, the network-based toolkit performed with no delay while running on a consumer-grade laptop, demonstrating that the toolkit is flexible enough to run on any platform compatible with Bro. Custom Bro scripts should work under all possible constraints in the ICS network because they are passive by design.

2. Explanation of Host-based Toolkit Functions

Figure 12 shows a use and compatibility matrix for this research’s host-based toolkit, illustrating all commands used and the operating system support for each command. The core utilities were used in all scripts and the individual commands beneath were only needed for individual adversary tactic identification scripts. Of note, WMIC is heavily used in this toolkit and while it is not included by default in OS versions prior to Windows XP, Microsoft released a package that allows it to function with legacy versions starting with Windows 95. Executing this package on remote machines requires some additional steps to ensure the software is running [71]. Any tools that require the installation of additional resource kit packages have been noted on the Figure 12 [72]. REG was used as a quick, compatible method to query the Windows registry for specific keys related to the eight adversary tactics within all scripts. The FIND command and the more demanding FINDSTR command when necessary were used to incrementally search for values and regular expression strings within the output of several other commands. WEVTUTIL was included for its ability to quickly isolate

relevant event log files, but its use is optional since the kit was designed with the expectation that WEVTUTIL would not be supported and that the target system's event logs would be insufficient.

Operating System	OS Name	95 / NT	98	ME	2000	XP	XP 64 / Server 2003	Vista / Server 2008	7	8 / Server 2012
	Version	4.0	4.1	4.9	5.0	5.1	5.2	6.0	6.1	6.2
Core Tools	WMIC	☑*	☑*	☑*	☑*	☑	☑	☑	☑	☑
	REG	☑*	☑*	☑*	☑*	☑	☑	☑	☑	☑
	FOR	☑	☑	☑	☑	☑	☑	☑	☑	☑
	FINDSTR	☑*	☑*	☑*	☑	☑	☑	☑	☑	☑
	FIND	☑	☑	☑	☑	☑	☑	☑	☑	☑
	WEVTUTIL							☑	☑	☑
Tactic Module-Specific Tools	ARP	☑	☑	☑	☑	☑	☑	☑	☑	☑
	IPCONFIG	☑	☑	☑	☑	☑	☑	☑	☑	☑
	NETSH				☑	☑	☑	☑	☑	☑
	NETSTAT	☑	☑	☑	☑	☑	☑	☑	☑	☑
	TYPE	☑	☑	☑	☑	☑	☑	☑	☑	☑
	SC				☑	☑	☑	☑	☑	☑
	SCHTASKS					☑	☑	☑	☑	☑
	BITSADMIN							☑	☑	☑
	DRIVERQUERY					☑	☑	☑	☑	☑
	TREE	☑	☑	☑	☑	☑	☑	☑	☑	☑
	DIR	☑	☑	☑	☑	☑	☑	☑	☑	☑
	ASSOC				☑	☑	☑	☑	☑	☑
	FTYPE				☑	☑	☑	☑	☑	☑
	NBTSTAT		☑	☑	☑	☑	☑	☑	☑	☑
	DOSKEY	☑	☑	☑	☑	☑	☑	☑	☑	☑
*Note: OS version requires installation of additional command-line utility package										

Figure 12. Host-based command usage and operating system compatibility

The additional command-line utilities used only in selected scripts are presented at the bottom of Figure 12 in the same order as the corresponding adversary tactic is described in this thesis. Most utilities are run with specific, tested command line parameters to ensure stability of hosts. The ICS field device script used ARP, IPCONFIG, and NETSH to extract MAC addresses from the local network, host machine, and in-range wireless devices, respectively. The external connection host-based

script used IPCONFIG, NETSTAT, and TYPE to display the DNS cache, correlate a host process to network behavior observed, and to output cookies from file paths determined by Windows version.

While registry persistence was checked entirely with the core REG command, the startup portion of the persistence script employed TYPE to output contents of each host's startup folders and SC to query services that automatically restart or trigger based on failure conditions. The scheduled task portion of the persistence script primarily used WMIC but also applied the SCHEDULETASK and BITSADMIN utilities to output their respective scheduled job data to a central report for identification of malicious dormancy and privilege escalation attempts.

The process injection and hijacking script mainly used the core WMIC tool's querying language as well as FIND to identify processes running from %TEMP% or %LOCALAPPDATA% and to compare processes and DLL imports with ICS artifact whitelists and blacklists. The process injection script also leveraged DRIVERQUERY to display details about loaded unsigned drivers as well as a consolidated list of certificates used for the drivers that were signed for easy analysis.

The file system sabotage script harnessed WMIC to parse the cim_datafile for compressed archives along with TREE and DIR with special parameters to check for unquoted executables, recently created binaries, and anomalous files in the RECYCLER. The file system sabotage script also used REG to query SOFTWARE\Classes and provide anomalous debugger and image file execution keys. The file system sabotage script combined results of ASSOC and FTYPE to output file extensions and their linked programs for anomalous extensions that do not exist in the baseline operating system.

The internal lateral movement script relied on WMIC to provide various shared resources such as printers that could span network zones. The same script also used NETSTAT to check potential services in use and the NBTSTAT tool to display cached historical NetBIOS connections.

The portable media script employed only the core utilities to extract currently- and previously-connected portable devices, extracting uniquely-identifying data from various registry entries to correlate their usage across all hosts. The script also extracted AutoRun settings from various forensic locations to determine portable media propagation risk.

The user behavior and remote access abuse script largely used WMIC and WEVTUTIL (if available) to first search for specific Event IDs if present, then query TerminalServices, RemoteConnectionManager, and LocalSessionManager for RDP artifacts in the absence of security event log data. The script was built with input parameters to whitelist certain time periods so as to narrow results to specified anomalous off-hours. The user behavior and remote access abuse script also used DOSKEY to extract explicit command-line usage from memory.

3. Concerning Findings

The network-based scripts to passively identify external communication paths from the ICS network proved to be valuable and reliable. External connections from the ICS network were quickly identified in the studied data that provided an understanding of how the network was configured, such as a company's interconnection of an ICS network with the DNS and network time protocol (NTP) servers on their business network in Figure 13. Other systems attempted outbound connections to the Internet over NetBIOS and other non-ICS protocol ports. In addition, pathways and network traffic fragments were identified between several devices and 143.127.102.40 in Cupertino, CA, Symantec's LiveUpdate server for virus definitions; this was further validated when the host-based scripts extracted securityresponse.symantec.com and liveupdate.symantecliveupdate.com from the DNS cache. A separate ICS network traffic sample showed systems attempting to connect to guru.avg.com and bguru.avg.cz to update the Anti-Virus Guard (AVG) antivirus product. In yet another sample data set, the host- and network-based scripts quickly revealed other attempts to product update websites for multiple ICS equipment vendors. Although the inconsistent attempts to external networks in this data were system misconfigurations and attempted product

updates, this analysis could have similarly identified beaconing malware and malicious external command and control attempts.

165	192.	.67		53	DNS			US	ESTABLISHED
102	192.	.11	143.127.102.40	443	HTTP/SSL	37.304199	-122.094597	Cupertino CA	US PATHEXISTS
87	192.	.230		123	NTP			US	ESTABLISHED
87	192.	.16	143.127.102.40	443	HTTP/SSL	37.304199	-122.094597	Cupertino CA	US PATHEXISTS
86	192.	.230		123	NTP			US	ESTABLISHED
49	192.	.245		514	unknown			US	ATTEMPTED
47	192.	.67		138	unknown			US	ATTEMPTED
42	192.	.70		137	NetBIOS			US	ATTEMPTED

Figure 13. Identified communication paths from ICS network

The real-time visualization of attempted and established external network connections from the ICS domain provided value once the automated offline analysis and packet capture (PCAP)-to-JSON conversion scripts were completed. The jVectorMap representation of one real-world ICS network’s external connection attempts is shown in Figure 14. When foreign IP addresses are included in the JSON data, the map was programmed to display a world map.

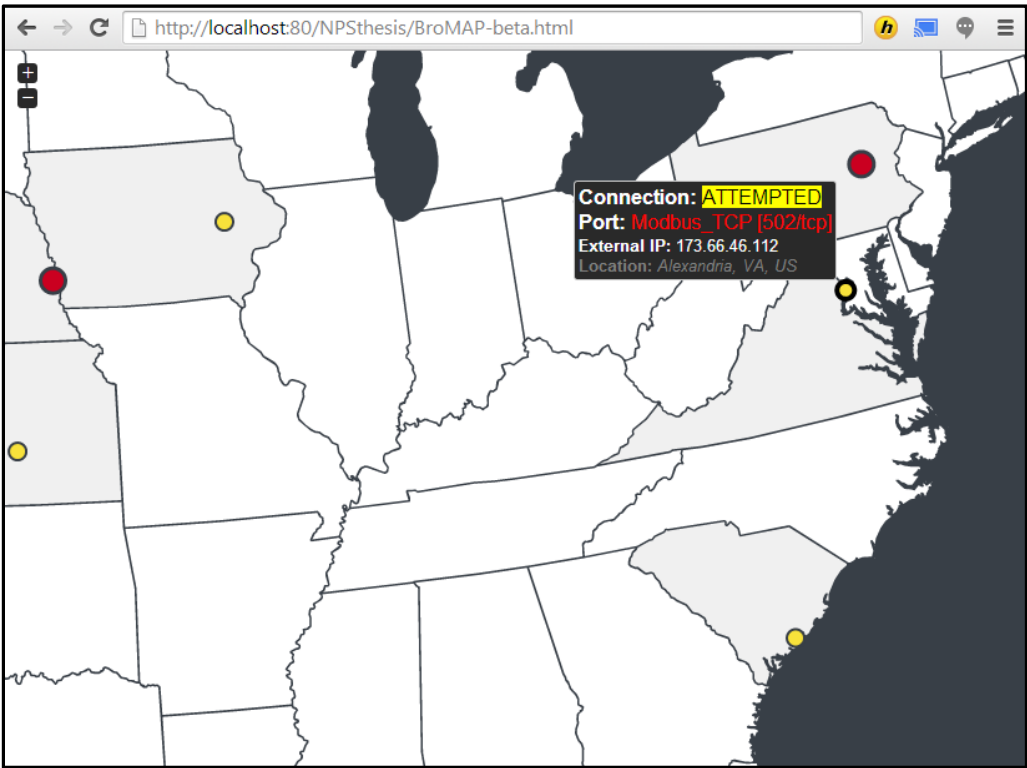
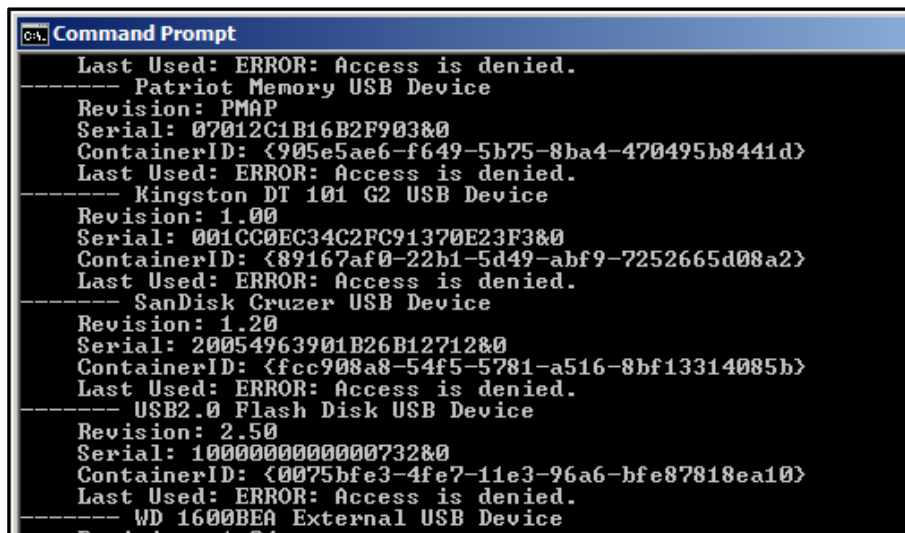


Figure 14. External connections output from this thesis’ network-based toolkit

HTTP protocol analysis revealed interesting characteristics unique to ICS networks. Built-in Bro IDS alerts were triggered because some traffic samples included invalid HTTP methods. In those cases, vendors had configured their product to use a [VENDORNAME]_POST method and used HTTP as a convenient transport protocol. Other analysis showed HTTP access to DLL files and posting values directly as variables (/[/vendor]/[vendor].dll?v=update). These ICS device HTTP traits will be added to the growing whitelist of atypical vendor implementations of traditional network protocols.

USB drive insertion data is of immediate interest during incident response and is critical for sites whose network security policies disallow portable media usage. Figure 15 shows the output of the host-based script when run against a test system with many USB drive insertions. In the prototype toolkit there are permission issues accessing the Properties subkey which contains dates that were originally going to be plotted on a timeline. This will be fixed in the next iteration, and the AutoRun flag and the user who accessed the drive will also be displayed to further assist security teams.



```
cmd Command Prompt
Last Used: ERROR: Access is denied.
----- Patriot Memory USB Device
Revision: PMAP
Serial: 07012C1B16B2F903&0
ContainerID: <905e5ae6-f649-5b75-8ba4-470495b8441d>
Last Used: ERROR: Access is denied.
----- Kingston DT 101 G2 USB Device
Revision: 1.00
Serial: 001CC0EC34C2FC91370E23F3&0
ContainerID: <89167af0-22b1-5d49-abf9-7252665d08a2>
Last Used: ERROR: Access is denied.
----- SanDisk Cruzer USB Device
Revision: 1.20
Serial: 20054963901B26B12712&0
ContainerID: <fcc908a8-54f5-5781-a516-8bf13314085b>
Last Used: ERROR: Access is denied.
----- USB2.0 Flash Disk USB Device
Revision: 2.50
Serial: 10000000000000732&0
ContainerID: <0075bfe3-4fe7-11e3-96a6-bfe87818ea10>
Last Used: ERROR: Access is denied.
----- WD 1600BEA External USB Device
Revision: 1.00
```

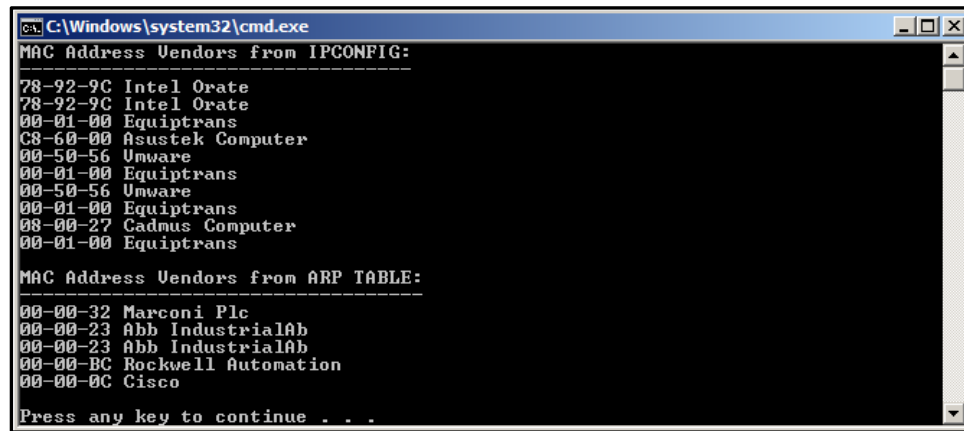
Figure 15. USB insertion script output on test system

The above results constitute suspicious data of moderate confidence. As more data is processed, such suspicious findings can be further separated into confirmed malicious findings in the form of a blacklist, and moderate-confidence suspicious

findings will be used to maintain a more flexible ICS behavioral greylist that may be only indicative of compromise when combined with other observed activity.

4. Valuable Discoveries

The initial version of this toolkit provided valuable configuration information about a particular site as well as a more general understanding of industry ICS network installments across multiple sets of data. Execution from both the command-line of the system and on live or historical network data provides new observational capabilities. For instance, Figure 16 shows remote querying of an endpoint to identify connected equipment. This host-based script was further improved to show in-range wireless access points and previously wireless connections, if any.



```
C:\Windows\system32\cmd.exe
MAC Address Vendors from IPCONFIG:
78-92-9C Intel Orate
78-92-9C Intel Orate
00-01-00 Equiptrans
C8-60-00 Asustek Computer
00-50-56 Umware
00-01-00 Equiptrans
00-50-56 Umware
00-01-00 Equiptrans
08-00-27 Cadmus Computer
00-01-00 Equiptrans

MAC Address Vendors from ARP TABLE:
00-00-32 Marconi Plc
00-00-23 Abb IndustrialAb
00-00-23 Abb IndustrialAb
00-00-BC Rockwell Automation
00-00-0C Cisco
Press any key to continue . . .
```

Figure 16. Host characteristics extracted from command-line utilities

Although host-based command-line device enumeration will be valuable in environments without historical network traffic captures, the network scripts created in Bro demonstrated far more utility and flexibility in this area. The network-based enumeration and correlation across several nodes is completely passive and device-agnostic. Bro is built around real-time generation of logs for immediate viewing and alerts, but its language supports a variety of outputs and data structures. Traditional stream-based output is supported in the ICS device-fingerprinting code created for this thesis, but the real utility of the toolkit is the comprehensive collection of device

information in scalable arrays and searchable sets of addresses that associates known device information together automatically, without having to further parse the output.

The script pulls MAC (physical) addresses and IP (logical) addresses from ARP requests and replies, and extracts MACs, IPs, and hostnames from DHCP request, inform, and discovery packets. MAC addresses were passively observed for 64% of devices on the LAN and 100% of the hardware vendors were identified from those MAC addresses. 100% of the internal device IP addresses were collected, which is to be expected for TCP/IP to properly function, and none were identified as dual-homed (a system having multiple network interface cards and thus multiple IP addresses per one MAC address). Only 12% of hostnames were observed in DHCP traffic, likely due to the prevalence of static IP assignments in critical networks. This was a known consideration during network toolkit creation, and so the script includes an event to trigger when a DNS address (type A) reply is to an internal IP address that does not yet have a known associated hostname and for which the corresponding query's subdomain substring is used for the hostname. Using both DHCP and internal DNS extraction, 41% of hostnames were identified. These hostnames have been removed Figure 17 to keep the processed data private. According to Bro's built-in operating system analyzer, the majority of operating systems in the data were outdated and unsupported versions of Microsoft Windows, validating the focus on those systems with the host-based kit.

Vendor	Count	%
Undetermined	42	36%
Hewlett-Packard	23	20%
Dell	14	12%
Vmware	11	9%
Cisco	8	7%
Moxa	3	3%
Cadmus Computer	2	2%
JK Micro	2	2%
Bristol Babcock	1	1%
Fortinet	1	1%
Juniper Networks	1	1%
Rockwell Automation	1	1%
Selectron	1	1%
Sixnet	1	1%
Stratus Computer De	1	1%
Termtek Computer Co	1	1%
Xyplex	1	1%
Yaesu Musen Co	1	1%
ZF Micro	1	1%
System Total	116	

Operating System	Count	%
Undetermined	71	61%
Windows 2000 SP4, XP SP 1+	34	29%
Windows 2000 SP2+, XP SP1, 98	5	4%
Windows XP SP1+, 2000 SP3	4	3%
Linux 2.6	2	2%
System Total	116	

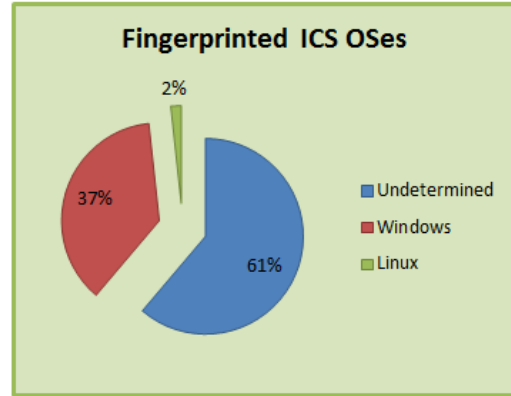


Figure 17. Summary of host characteristics extracted from network data

The toolkit extends the built-in fingerprinting with some initial ICS protocol and supervisory device trait-matching. For instance, based on publicly-available vulnerability data released for Wonderware's SuiteLink service [73], any Windows OS host listening on UDP port 5413 with connections to one or more systems that speak other ICS protocols is flagged by the script as likely a WonderWare InTouch HMI. This rough ICS device cataloging capability should grow as more artifacts are ingested and processed.

For proper comparison, 24-hour periods of Modbus and DNP3 network traffic captures from several critical infrastructure sectors were analyzed and correlated. One of this thesis' Bro scripts attempted to identify ICS protocol abnormalities while another script attempted to fingerprint devices' specific roles in the ICS network based on their register values and their function and exception codes. The first script's analysis indicated that the ingested data did not contain any protocol convention anomalies or suspicious ICS function code usage, which may indicate malicious tampering or field device modification. All examined ICS protocol traffic contained only register reads and writes, which was confirmed through packet analysis outside of the toolkit, so the

extraction of potentially malicious activity based on anomalous protocol events was not tested.

Analyzing the same source data, the second Bro script's output did not indicate a significant correlation between ICS protocol function code patterns or register-boundary values and the field device's role (e.g., HMI, RTU, PLC). This may also be a result of the limited ICS device functions observed. Frequency analysis of observed Modbus and DNP3 function codes is provided in Figure 18. Across all samples, slave field devices had an average of 5.5 registers. Register ranges varied significantly even on a single device, as in one case a device's register only changed by within a range of one across a 24-hour period but the device's other register had a range of 65,369. The average low boundary value for registers was 8,594 and the average high boundary was 37,248, but the protocols are device-agnostic so more meaningful anomalies or further indications of malicious behavior could not be extracted from the field devices' numeric register-values without knowing the specific processes they controlled. There are likely values for field device register limits and metrics so alerts could be created based on manually-input expectations. Alternately, future iterations of the Bro script may record register changes as a percentage of the current values. The extraction and correlation of ICS protocol functions did however support the assertion that ICS networks have a high signal-to-noise ratio since anomalous function codes, exception codes, and malformed packets should be significantly distinguishable if present in captured data.

Protocol Function (Code)	DNP3	Modbus
DNP3 Confirm (0)	0.02%	0.00%
DNP3 Read (1)	99.98%	0.00%
Modbus Read Coils (1)	0.00%	56.26%
Modbus Read Holding Registers (3)	0.00%	25.90%
Modbus Read Input Registers (4)	0.00%	13.57%
Modbus Write Single Coil (5)	0.00%	3.01%
Modbus Write Multiple Registers (16)	0.00%	1.26%
Grand Total	100.00%	100.00%

Figure 18. ICS protocol frequency analysis in dataset

C. TOOLKIT LIMITATIONS

This section covers several weaknesses in the current version of the methodology, including examples of tool-specific inadequacies and a survey of observed type I and type II errors in the automated analysis.

1. Toolkit Deficiencies

One significant limitation of the approach is its inability to calculate cryptographic hashes to verify authenticity of files due to the absence of built-in Windows command-line hashing. Since firmware is not commonly digitally signed for ICS devices, hashes should be computed to compare against manufacturer data provided in documentation or online. Several community tools are available and Microsoft released a command-line hashing utility (FCIV.exe) in 2004, but it is not built into the Windows OS versions examined. A PowerShell script can likely accomplish cryptographic hashing but that limits compatibility and a scripted cryptographic routine may overly tax the limited resources. Bro's file framework is impressive and can passively categorize, extract, and hash all files sent in packets, so modification and extension of Bro's file framework will aid future firmware analysis functionality.

Several studied adversary attack techniques only produced accurate forensic artifacts when the same command-line utilities used by an attacker were also used to query the host by a responder. This suggests the need to investigate command-line utility outputs before claiming full coverage. For example, within the task-scheduling persistence identification script, simulated malicious tasks for privilege escalation and reboot persistence created on a Windows XP host with the AT command were only revealed by WMIC, and tasks created with SCHEDULETASKS command were only revealed by querying SCHEDULETASKS. On Windows Vista and up, a third built-in tool, BITSADMIN, allows scheduling of malicious tasks that are only revealed by listing from BITSADMIN and no other tools, as illustrated in Figure 19.

```

C:\Windows\system32>bitsadmin /list /verbose
BITSADMIN version 3.0 (7.5.7681)
BITS administration utility.
(C) Copyright 2000-2006 Microsoft Corp.

BITSAdmin is deprecated and is not guaranteed to be available in future versions
of Windows.
Administrative tools for the BITS service are now provided by BITS PowerShell co
ndlets.

GUID: {D50F86B0-153E-4372-AF16-...} DISPLAY: 'SCADAhackToolz'
TYPE: DOWNLOAD STATE: SUSPENDED OWNER:
PRIORITY: NORMAL FILES: 0 / 1 BYTES: 0 / UNKNOWN
CREATION TIME: 5/20/2014 5:26:28 PM MODIFICATION TIME: 5/20/2014 5:36:55 PM
COMPLETION TIME: UNKNOWN ACL FLAGS:
NOTIFY INTERFACE: UNREGISTERED NOTIFICATION FLAGS: 0
RETRY DELAY: 600 NO PROGRESS TIMEOUT: 1209600 ERROR COUNT: 0
PROXY USAGE: PRECONFIG PROXY LIST: NULL PROXY BYPASS LIST: NULL
DESCRIPTION:
JOB FILES:
0 / UNKNOWN WORKING http://127.0.0.1/hadfirmware.ful -> C:\Siemens\Sinat
leCC\current.ful
NOTIFICATION COMMAND LINE: 'C:\Windows\System32\cmd.exe' 'C:\Windows\System32\notepad.exe'
BITSADMIN PERSISTENCE:
owner MIC integrity level: MEDIUM
owner elevated? false
PeerCaching Flags
Enable download from peers :false
Enable serving to peers :false
CUSTOM HEADERS: NULL
Listed 1 job(s).
C:\Windows\system32>

C:\Windows\system32>UMIC JOB GET command, daysofweek, jobstatus, runrepeatedly
Command: daysofweek jobstatus runrepeatedly
C:\Windows\System32\notepad.exe 15 Success TRUE

C:\Windows\system32>SCHTASKS /QUERY /U /FO LIST /TN SCADAconnect
Folder: \
HostName:
TaskName: \SCADAconnect
Next Run Time: 6/1/2014 5:18:00 PM
Status: Ready
Logon Mode: Interactive only
Last Run Time: N/A
Last Result: 1
Author:
Task To Run: C:\Windows\System32\cmd.exe /c C:\Windows\S
system32\ntstsc.exe
Occur in: N/A
Comment: N/A
Scheduled Task State: Enabled
Idle Time: Disabled
Power Management: Stop On Battery Mode, No Start On Battery
Run As User:
Delete Task If Not Rescheduled: Enabled
Stop Task If Runs X Hours and X Mins: 72:00:00
Schedule: Scheduling data is not available in this f
Format:
Schedule Type: Monthly
Start Time: 5:18:00 PM
Start Date: 5/20/2014
End Date: N/A
Days: First SUN
Months: Every month
Repeat: Every: Disabled
Repeat: Until: Time: Disabled
Repeat: Until: Duration: Disabled
Repeat: Stop If Still Running: Disabled

```

Figure 19. Three command-line tool queries with mutually-exclusive results

Although this host script does query each service and identifies jobs scheduled from all methods, the output is difficult to parse and thus a more reliable, central location should be used to find these tasks. Forensic locations such as %windir%\Tasks\SchedLgU.txt for Windows XP and below, and Microsoft-Windows-TaskScheduler%40Operational.evtx for Vista and above, may centrally store this data for all tools and the scheduled task persistence identification script can extract data from those locations.

Separating malicious access from trusted access was difficult in the ICS environment despite the high signal-to-noise ratio. While the toolkit reliably distinguished automated device activity from human-initiated activity, further separating trusted user behavior from malicious attacker behavior was challenging with limited historical visibility and inconsistent logging on the systems analyzed. Some promising data was found outside of command-line tools and parsed event logs, such as within the *.rdp files on the host and within parsed remote access network traffic, but the initial scripts were unable to automatically extract that data in a meaningful way that applied to multiple sites.

Similarly, malicious internal lateral movement was difficult to identify in this environment due to the number of accounts with administrative privileges and ICS vendors' pervasive usage of server message block (SMB) and NetBIOS for access to

shared resources. This should diminish with the development of more robust vendor-specific whitelists.

2. False Positive Errors

Type I errors, or false positives, occur when non-malicious is identified as malicious activity. The collection of large datasets without honed capabilities to analyze the data resulted in a high number of false positives.

False positives were more likely to occur when only host or network data was available. Ideally, toolkits should try to determine where data went that left a host or what process was running that was using a suspicious protocol. In these experiments, some host-based activity that was flagged as malicious was SCADA engineers moving laptops between network zones. It was assumed that a host remained in an environment, so laptops moved between zones generated several suspicious traits, such as using many fully-qualified domain names that did not actually represent external attempts from within the ICS network. But since connecting laptops to and from the ICS domain is not advisable, these may not be considered false positives.

Within one data set, an indication of external information exfiltration over an ICS protocol, OASys DNA, was actually a Type I error that resulted from incorrect programming. An external connection identification Bro script did not originally include 255.0.0.3 in its whitelist, a reserved multicast address. In another set of data analyzed, a router was configured to send a response back to attempts to connect out of the ICS network. The toolkit's connection state logic has since been reviewed and the IP subnet whitelist now correctly reflects all intended requests for comments (RFCs) from the Internet Engineering Taskforce (IETF).

Within the network-based toolkit results, more activity was observed than expected with Telnet, NetBIOS, SNMP, and HTTP protocols. Specifically, many ICS systems were observed using embedded HTTP functionality for diagnostics and monitoring. The analysis of unexpected HTTP user-agent strings requires a more comprehensive understanding of the ICS software that uses HTTP, or the use of filtering

by unique properties like session duration, so that fewer anomalies are highlighted to the toolkit's user.

3. False Negative Analysis

A Type II error, or a false negative, is when malicious activity existed but was not detected. Because none of the data included a confirmed compromise, errors of this category were difficult to find. It is important, to note that, when searching for malicious attackers, the old criminal profiling aphorism “the absence of evidence is not the evidence of absence” applies.

One possible design flaw that could lead to false negatives is within the geolocation and visualization of external connections. Several samples included external attempted and established connections to cellular data devices. In one instance, a SCADA server was communicating with several dozen Verizon devices with no uniform IP address assignment, which could reflect sites replacing old direct dial-up access with cellular data connections. Geolocation will never be able to place these devices accurately on a map since they do not have set locations and are assigned in pools by mobile providers. Although separate online whois queries can be conducted and collated for some assistance, an adversary could blend into this access pathway or maintain persistence by beaconing out of an ICS network to an adversary-owned mobile device from the same provider.

Additional false negatives may have occurred in the studied data due to the lack of built-in Bro support for ICCP, OPC, and some other observed proprietary ICS protocols. However, these protocols communicate in cleartext like many other supported protocols (e.g., Telnet, FTP, MySQL, and HTTP), so more work needs to be done on the network toolkit to include the parsing of cleartext ICS protocols.

VI. CONCLUSIONS AND FUTURE WORK

A. IMPACT

This thesis described the need for a forensic capability tailored to industrial control systems. It then described and tested several techniques for identifying and analyzing potential malicious activity and adversary access pathways within ICS networks. This resulted in the creation of several tools to be used for live production ICS forensics and suggested some approaches to building high-confidence indicators of unauthorized access. ICS networks do provide good signal-to-noise ratios for behavioral anomaly detection. The need for tailored tools to identify potentially malicious access pathways has been suggested by this work and the feasibility of such tools has been established with the prototype host- and network-based toolkit.

Security assessment and incident response teams can use the approach to analyze critical devices in an environment with minimal irrelevant data, then trace those findings back into the business network instead of the other way around. The tools should assist in making more confident intrusion judgments when malicious activity is suspected in the ICS environment. With the careful selection of compatible tools, and a technique that is passive and forensically cautious, these methods can be easily incorporated into many organizations' security processes.

B. FUTURE WORK

Due to the relatively long life cycle of systems in the ICS domain and the current compatibility with Windows 8 systems, this technique is expected to be functional for a while on Microsoft operating systems. However, Linux compatibility is needed and work has already begun on translating the host-based examination commands for Linux. The WMI core application has been ported to Linux, which presents a desirable host-platform shift since Bro runs on Linux. If the host and network tools are executed from the same system, compatibility should expand and the ability to cross-reference host and network data should improve greatly. Adding agentless querying of real-time operating systems such as VxWorks, Windows CE, QNX and embedded Linux should also be pursued.

The host-based toolkit can be run on each host, transferring the HTML-formatted output to a write-once CD or other protected media. The toolkit can also be run on multiple hosts and centrally-collected, but this has only been tested on small, closed networks, and already is generating significant amounts of data. If both host and network data can be centrally analyzed, the data will require improved structures to ensure performance is still acceptable. This approach of central collection and analysis of host and network data allows additional features such as rootkit identification to become possible by comparing host and network data. For example, if a port is communicating in network traffic but not showing on a host's netstat querying, a rootkit may be hiding it from the Windows API.

The migration to a NoSQL document database for forensic data management would increase scalability and reduce endpoint processing cycles even further. Alternately, Bro may be capable of ingesting the host-based forensic output instead of a NoSQL database and it could be used as a source-agnostic state machine to process events. More interactive mapping of specifically-filtered data could be valuable. Examples are the data-driven documents (D3) JavaScript framework to depict relationships between collected artifacts, the portrayal of host anomaly timelines, and visual incident triage beyond pure geospatial representation.

Future toolkit development should pursue the inclusion of network policy checklists for each site to determine the logic used to find anomalies. To aid in understanding potential adversary pathways, the toolkit should also include the collection of or manual entry of network interconnect information, such as firewall logs, between zones. It may be possible to identify the domain interconnection type from passive network traffic already collected, since mechanisms such as data diodes have unique communication patterns.

Additional offline databases of security information could be incorporated beyond geolocation data, vendor information, and ICS protocol identification. Bro now has an offline autonomous system number (ASN) lookup function to determine border gateway protocol (BGP) routing and Internet Service Provider (ISP) information.

The malicious activity blacklists and whitelists used were relatively small and should be expanded. The total number of ICS-specific elements within the current lists is less than 200 as collected from observed ICS hosts in industry and online documentation (Figure 20). ICS hardware vendors could be contacted directly as an authoritative source to assist in compiling attribute whitelists for their equipment.

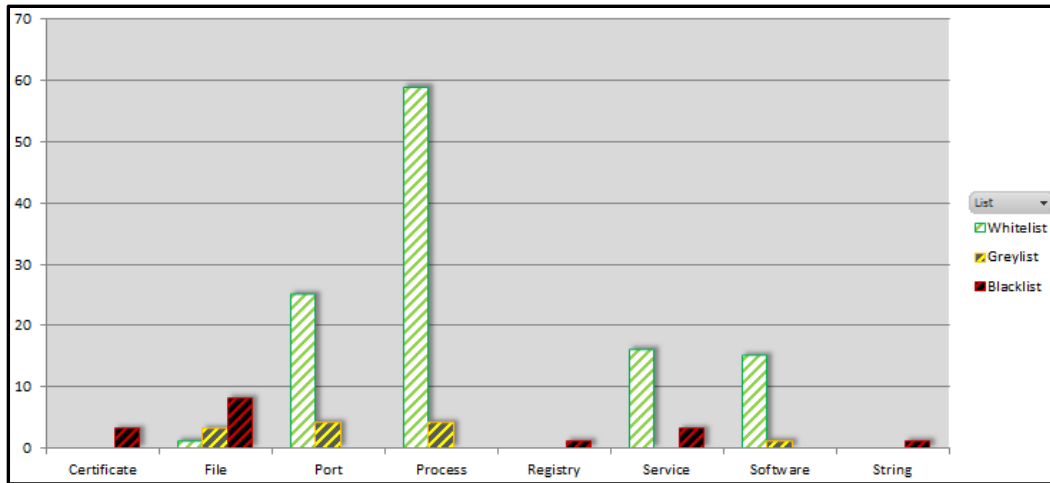


Figure 20. Content distribution of initial ICS-specific whitelist

Future testing of the toolkit should be conducted on data that includes known malicious activity attempts. To facilitate this, development of ICS network honeypots should be pursued with tools like Conpot, released in 2013 by the HoneyNet project to simulate ICS networks [74].

C. CONCLUDING REMARK

As the increasingly connected systems that power modern society's critical infrastructure are subject to more scrutiny and attacks, those systems' owners and operators will need novel techniques for assessing and defending their ICS networks. The methodology presented in this thesis has been tested and the supplementary toolkit has demonstrated value. The ICS security community would benefit from further funding and development of an incident response toolkit to assist in securing our Nation's critical assets.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

- [1] B. Salem et al., "Brief history and use of chemical warfare agents in warfare and terrorism," *Chemical Warfare Agents: Chemistry, Pharmacology, Toxicology, and Therapeutics*, p. 2, 2008.
- [2] W. Abrams, "Malicious control system cyber security attack case study—Maroochy Water Services, Australia," The MITRE Corporation, McLean, VA, 2008.
- [3] M. Slay, "Lessons learned from the Maroochy Water breach," *Critical Infrastructure Protection, Post-Proceedings of the First Annual IFIP Working Group 11.10 International Conference on Critical Infrastructure Protection, Dartmouth College, Hanover, New Hampshire, USA*, March 19–21, 2007.
- [4] C. Falliere, "W32.Stuxnet dossier whitepaper," Symantec Corporation, 2011.
- [5] DHS Industrial Control Systems Cyber Emergency Response Team, "ICS-CERT Monitor: April/May/June 2013," 27 June 2013. [Online]. Available: https://ics-cert.us-cert.gov/sites/default/files/Monitors/ICS-CERT_Monitor_Apr-Jun2013.pdf. [Accessed April 2014].
- [6] DHS Industrial Control Systems Cyber Emergency Response Team, "Recommended practice: Creating cyber forensics plans for control systems," August 2008. [Online]. Available: https://ics-cert.us-cert.gov/sites/default/files/recommended_practices/Forensics_RP.pdf.
- [7] P. Taveras, "SCADA live forensics: Real time data acquisition process to detect, prevent or evaluate critical situations," in *1st Annual International Interdisciplinary Conference*, Azores, Portugal, 2013.
- [8] P. M. Hill, "Process control in the HPI: A not-so-sentimental journey," 1 July 2012. [Online]. Available: <http://www.hydrocarbonprocessing.com/Article/3050923/Process-control-in-the-HPI-A-not-so-sentimental-journey.html>.
- [9] R. McClanahan, "SCADA and IP: Is network convergence really here?," *IEEE Industry Applications Magazine*, pp. 29–36, March 2003.
- [10] A. W. Schwartz, "Control system devices: Architectures and supply channels overview," Sandia National Laboratories, Albuquerque, NM, 2010.

- [11] Centre for the Protection of National Infrastructure (CPNI), “Securing the move to IP-based SCADA/PLC networks,” 2011.
- [12] W. Shaw, *Cybersecurity for SCADA Systems*, PennWell Corporation, 2006.
- [13] H. Kozushko, “Intrusion Detection: Host-Based and Network-Based Intrusion Detection Systems,” 11 September 2003. [Online]. Available: <http://infohost.nmt.edu/~sfs/Students/HarleyKozushko/Papers/IntrusionDetectionPaper.pdf> [Accessed 18 June 2014].
- [14] J. Lowe, “The myths and facts behind cyber security risks for industrial control systems,” in *Proceedings of the VDE Kongress*, 2004.
- [15] K. Stouffer et al., “Special publication 800-82: Guide to industrial control systems (ICS) security,” National Institute of Standards and Technology, Gaithersburg, MD, 2011.
- [16] C. Wueest, “Targeted attacks against the energy sector,” Symantec Security Response, Mountain View, CA, 2014.
- [17] V. Ijure, “Security issues in SCADA networks,” in *Computers and Security*, 2006, pp. 498-506.
- [18] E. W. Clarke, *Practical Modern SCADA Protocols: DNP3, 60870.5 and Related Systems*, Burlington, MA: Newnes, 2004.
- [19] “Modbus frequently asked questions,” [Online]. Available: <http://www.modbus.org/faq.php>.
- [20] DNP3.org, “DNP3: A distributed network protocol primer,” 2005. [Online]. Available: <http://www.dnp.org/AboutUs/DNP3%20Primer%20Rev%20A.pdf>. [Accessed March 2014].
- [21] ODVA, “Open DeviceNet Vendors Association (ODVA) EtherNet/IP (Ethernet Industrial Protocol),” [Online]. Available: <http://www.odva.org/default.aspx?tabid=67>. [Accessed April 2014].
- [22] Symantec, “SCADA vulnerability trends,” 2012. [Online]. Available: http://www.symantec.com/threatreport/topic.jsp?aid=scada_vulnerabilities&id=vulnerability_trends [Accessed 28 April 2014].

- [23] North American Electric Reliability Corporation, “Control systems security working group (CSSWG),” 2013. [Online]. Available: <http://www.nerc.com/comm/CIPC/Pages/Control%20Systems%20Security%20Working%20Group%20CSSWG/Control-Systems-Security-Working-Group-CSSWG.aspx> [Accessed June 2014].
- [24] D. Dzung et al., “Security for industrial communication systems,” *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1152–1177, 2005.
- [25] M. Cooksley, “Introduction to network security,” Belden Industrial Ethernet Infrastructure Design Seminar, Philadelphia, PA, 2012.
- [26] DHS Industrial Control Systems Cyber Emergency Response Team, “ICS-CERT Monitor: January - April 2014,” 16 May 2014. [Online]. Available: http://ics-cert.us-cert.gov/sites/default/files/Monitors/ICS-CERT_Monitor_%20Jan-April2014.pdf. [Accessed 21 May 2014].
- [27] A. Townsend, *Smart Cities: Big Data, Civic Hackers, and the Quest for a New Utopia*, New York: W. W. Norton & Company, 2013, pp. 268–269.
- [28] R. M. Lee, *SCADA and Me: A Book for Children and Management*, Birmingham, MI: IT-Harvest Press, 2013.
- [29] S. Adey, “The hunt for the kill switch,” *IEEE Spectrum*, May 2008.
- [30] J. Peerenboom, “Analyzing infrastructure interdependencies: Overview of concepts and terminology,” Argonne National Laboratory, Argonne, IL, 2007.
- [31] K. Rinaldi et al., “Identifying, understanding, and analyzing critical infrastructure interdependencies,” *IEEE Control Systems Magazine*, 2001.
- [32] National Transportation Safety Board, “Pipeline accident report: Pipeline rupture and subsequent fire in Bellingham, Washington,” Washington, DC, 2002.
- [33] K. Poulsen, “Software bug contributed to blackout,” 11 February 2004. [Online]. Available: <http://www.securityfocus.com/news/8016> [Accessed 23 May 2014].
- [34] U.S. Senate Committee on Commerce, Science, and Transportation, “Isolation of condensate demineralizer system and subsequent plant trip while testing software change (Hatch),” U.S. Government Printing Office, Washington, 2008.

- [35] J. Meserve, "Video: Staged cyber attack reveals vulnerability in power grid," CNN, 9 September 2007. [Online]. Available: <http://www.cnn.com/2007/U.S./09/26/power.at.risk/> [Accessed 15 March 2014].
- [36] Schweitzer Engineering Laboratories, Inc., "Mitigating the Aurora vulnerability with existing technology," in *36th Annual Western Protective Relay Conference*, Spokane, WA, 2009.
- [37] Schweitzer Engineering Laboratories, Inc., "Myth or reality: Does the Aurora vulnerability pose a risk to my generator?," in *IEEE Conference for Protective Relay Engineers*, College Station, TX, 2011.
- [38] S. Dunlap, "Timing-based side channel analysis for anomaly detection in the industrial control systems environment," Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2013.
- [39] E. Chien, "Stuxnet: A breakthrough," 16 November 2010. [Online]. Available: <http://www.symantec.com/connect/blogs/stuxnet-breakthrough> [Accessed 3 February 2014].
- [40] S. Bosworth, *Computer Security Handbook*, New York, NY: John Wiley & Sons Inc., 2002.
- [41] DHS Industrial Control Systems Cyber Emergency Response Team, "ICS-CERT Monthly Monitor: September 2012," September 2012. [Online]. Available: https://ics-cert.us-cert.gov/sites/default/files/ICS-CERT_Monthly_Monitor_Sep2012_2.pdf. [Accessed April 2014].
- [42] K. Kambic et al., "Crude Faux: An analysis of cyber conflict within the oil & gas industries," Purdue University, West Lafayette, 2013.
- [43] C. Bronk et al., "The cyber attack on Saudi Aramco," in *Survival*, 2013, pp. 81-96.
- [44] C. D. Schuett, "Programmable logic controller modification attacks for use in detection analysis," Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2014.
- [45] S. Kuvshinkova, "SQL Slammer worm lessons learned for consideration by the electricity sector," North American Electricity Reliability Council (NERC), Princeton, NJ, 2003.

- [46] Tofino Security, "Cyber security incident case profile: Daimler Chrysler," [Online]. Available: http://www.tofinosecurity.com/sites/default/files/CP-104-Case_Profile-Daimler_Chrysler-rev1.pdf. [Accessed 25 April 2014].
- [47] R. Esposito, "Hackers penetrate water system computers," October 2006. [Online]. Available: http://abcnews.go.com/blogs/headlines/2006/10/hackers_penetra/.
- [48] DHS Industrial Control Systems Cyber Emergency Response Team, "ICS-CERT October/November/December 2012 Monitor," 2 January 2013. [Online]. Available: http://ics-cert.us-cert.gov/sites/default/files/Monitors/ICS-CERT_Monitor_Oct-Dec2012.pdf. [Accessed April 2014].
- [49] Japanese Atomic Energy Agency, "Incident disclosure," 6 January 2014. [Online]. Available: https://www.jaea.go.jp/04/monju/EnglishSite/4_Press%20release/140106_press_release.pdf.
- [50] Context Information Security, "Context threat advisory: The Monju incident," 27 January 2014. [Online]. Available: http://contextis.co.uk/research/blog/CTI_TA-monju-incident/ [Accessed 19 March 2014].
- [51] DHS Industrial Control Systems Cyber Emergency Response Team, "Common cybersecurity vulnerabilities in industrial control systems," May 2011. [Online]. Available: http://ics-cert.us-cert.gov/sites/default/files/documents/DHS_Common_Cybersecurity_Vulnerabilities_ICS_2010.pdf.
- [52] E. Byres et al., "Security incidents and trends in SCADA," in *The Industrial Ethernet Book*, 2007, pp. 12-20.
- [53] D. E. Denning, "Cyberterrorism: The logic bomb versus the truck bomb," *Global Dialogue*, vol. II, no. 4, pp. 29-37, 2000.
- [54] Department of Justice, "Press release: Juvenile computer hacker cuts off FAA tower at regional airport," Boston, MA, 1998.
- [55] D. Powner, "Critical infrastructure protection: Multiple efforts to secure control systems are under way, but challenges remain," GAO-07-1036, 2007.
- [56] *United States of America v. Mario Azar*, 2009.

- [57] B. Gertz, "The cyber-dam breaks," 1 2013 May. [Online]. Available: <http://freebeacon.com/national-security/the-cyber-dam-breaks/> [Accessed June 3 2014].
- [58] Sandia National Laboratories, "Penetration testing of industrial control systems," March 2005. [Online]. Available: http://energy.sandia.gov/wp/wp-content/gallery/uploads/sand_2005_2846p.pdf [Accessed 19 March 2014].
- [59] Tofino Security, "Cyber security incident case profile: TVA/Browns Ferry," [Online]. Available: http://www.tofinosecurity.com/sites/default/files/CP-101-Case_Profile-Browns_Ferry-rev1.pdf. [Accessed 25 April 2014].
- [60] Digital Bond, Inc., "NERC CIP scan policies," [Online]. Available: <http://www.digitalbond.com/tools/bandolier/nerc-cip-scan-policies/> [Accessed 12 June 2014].
- [61] General Electric, "GE Fanuc support: Product work-arounds," [Online]. Available: http://support.ge-ip.com/support/resources/sites/GE_FANUC_SUPPORT/content/live/KB/8000/KB8854/en_U.S./Product%20Work-arounds.doc. [Accessed 2 May 2014].
- [62] Siemens AG, "ES - SIMATIC Manager product information and checking compatibility," 20 January 2012. [Online].
- [63] Centre for the Protection of National Infrastructure (CPNI), "Process control and SCADA security: Good practice guide," 2008. [Online]. Available: http://www.cpni.gov.uk/documents/publications/2008/2008031-gpg_scada_security_good_practice.pdf.
- [64] DHS Industrial Control Systems Cyber Emergency Response Team, "Recommended practice: Developing an industrial control systems cybersecurity incident response capability," October 2009. [Online].
- [65] U.S. Department of Energy, "21 steps to improve cyber security of SCADA networks."
- [66] Symantec Security Response, "W32.Disttrack technical details," 16 August 2012. [Online]. Available: http://www.symantec.com/security_response/writeup.jsp?docid=2012-081608-0202-99&tabid=2. [Accessed 3 March 2014].
- [67] M. Sully, "The deconstruction of the Mariposa botnet," February 2010. [Online]. Available: http://defintel.com/docs/Mariposa_White_Paper.pdf [Accessed 28 April 2014].

- [68] M. Russinovich, “Analyzing a Stuxnet infection with the Sysinternals tools,” 30 March 2011. [Online]. Available: <http://blogs.technet.com/b/markrussinovich/archive/2011/03/30/3416253.aspx>.
- [69] K. Poulsen, “Slammer worm crashed Ohio nuke plant network,” SecurityFocus, 19 August 2003. [Online]. Available: <http://www.securityfocus.com/news/6767>. [Accessed 28 May 2014].
- [70] Microsoft Corporation, “SetPriority method of the Win32_Process class,” [Online]. Available: [http://msdn.microsoft.com/en-us/library/aa393587\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/aa393587(v=vs.85).aspx). [Accessed 20 May 2014].
- [71] G. Williams, “Windows Management Instrumentation (WMI) support page,” Williams Technology Consulting Services, 12 July 2001. [Online]. Available: <http://www.wtcs.org/snmp4tpc/wmi.htm>. [Accessed 29 May 2014].
- [72] R. Van der Woude, “Reading NT’s registry with reg.exe,” Rob van der Woude’s Scripting, 4 March 2011. [Online]. Available: <http://www.robvanderwoude.com/ntregistry.php>. [Accessed 29 May 2014].
- [73] National Institute of Standards and Technology (NIST), “National cyber awareness system - Vulnerability summary for CVE-2008-2005,” DHS U.S.-CERT National Vulnerability Database, 6 May 2008. [Online]. Available: <http://web.nvd.nist.gov/view/vuln/detail?vulnId=2008-2005>. [Accessed 27 May 2014].
- [74] L. Rist, “Introducing Conpot,” The Honeynet Project, 11 May 2013. [Online]. Available: <https://www.honeynet.org/node/1047>. [Accessed 29 May 2014].
- [75] V. Paxon, “Bro: a system for detecting network intruders in real-time,” in *Proceedings of the 7th USENIX Security Symposium*, San Antonio, TX, 1998.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
Ft. Belvoir, Virginia
2. Dudley Knox Library
Naval Postgraduate School
Monterey, California